NASA CONTRACTOR REPORT

NASA CR-24

NASA CR-2492

ODIN: OPTIMAL DESIGN INTEGRATION SYSTEM

C. R. Glatt and D. S. Hague

Prepared by
AEROPHYSICS RESEARCH CORPORATION
Hampton, Va. 23666
for Langley Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1

1. Report No. NASA CR-2492	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle ODIN: Optimal Design Integration System		5. Report Date February [illegible]
		6. Performing Organization Code
7. Author(s) C. R. Glatt and D. S. Hague		8. Performing Organization Report No.
		10. Work Unit No.
9. Performing Organization Name and Address Aerophysics Research Corporation P. O. Box 7007 Hampton, Virginia 23666		11. Contract or Grant No. NAS1-12008
		13. Type of Report and Period Covered Contractor Report
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code

15. Supplementary Notes

Final Report

16. Abstract

The report provides a summary of the Optimal Design Integration (ODIN) System as it exists at Langley Research Center. A discussion of the ODIN System, the executive program and the data base concepts are presented. Two examples illustrate the capabilities of the system which have been exploited. Appended to the report are a summary of abstracts for the ODIN library programs and a description of the use of the executive program in linking the library programs.

17. Key Words (Suggested by Author(s)) Design, synthesis, ODIN, data base, design simulation, multiple programming		18. Distribution Statement Unclassified - Unlimited STAR Category 02	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 96	22. Price* $4.75

PREFACE

This report describes the Optimal Design Integration (ODIN) system in operation at Langley Research Center (LRC) at the completion of the contract NAS1-12008, "Expansion and Extension of the ODIN/RLV Computer Program." The study was carried out in the period from November, 1972, through November, 1973 with funds provided by the National Aeronautics and Space Administration, Langley Research Center, Space Systems Division, Vehicle Analysis Branch. The report specifically describes the ODIN system at LRC but generally applies to other systems which have been installed at the NASA Ames Research Center facility, the USAF Flight Dynamics Laboratory facility and the NASA Johnson Spacecraft Center facility.

The ODIN system was initially developed during 1971 and 1972 under parallel contracts with the National Aeronautics and Space Administration, LRC (NAS1-10692) and the USAF Flight Dynamics Laboratory (F33615-71-C-1480). The original system was developed for CDC 6000 series computer but the system has since been converted to the Univac 1100 series computer under contract (NAS9-12829) to NASA Johnson Spacecraft Center.

The approach to the research study was somewhat novel in that the combined efforts of the Aerophysics Research Corporation (ARC) personnel and the NASA Langley Research Center technical staff were utilized to accomplish the objectives. ARC provided the expansions, extension and consultation with regard to the ODIN library and executive programs. LRC provided the technical staff to establish and maintain ODIN design simulations in support of current NASA research activities. Numerous applications of the ODIN system have been accomplished by NASA personnel, two of which are presented.

CONTENTS

	Page
SUMMARY	1
INTRODUCTION	2
THE ODIN SYSTEM	5
The Program Library	5
The Executive Program	11
Data Base	14
APPLICATIONS	22
Orbiter Wing Design Study	24
Orbiter Landing Skin Temperature Study	29
CONCLUDING REMARKS	36
REFERENCES	38
APPENDIX A: BASIC ODIN PROGRAM ELEMENTS	41
ABLATOR: One Dimensional Analysis of the Transient Response of Thermal Protection Systems	45
ABNORM: Abnormal End Program	45
ACMOTAN: Linear Aircraft Motion Analysis	45
AESOP: Automated Engineering and Scientific Optimization Program	46
AFSP: Automated Flutter Solution Procedure	46
AIRFOIL: A Program for Generating Geometric and Aerodynamic Characteristics of Airfoil Sections	46
ATOP: Atmospheric Trajectory Optimization	47
AUTOLAY: A Program for Automatically Constructing Overlayed Computer Programs	47
CCLINK: Control Card Linkage Program	47
CCSAVE: Control Card Data Base Save Procedure	48
CHGABEN: Abnormal End Procedure	48
CHGDLOG: ODINEX Execution Procedure	48
COAP: Combat Optimization and Analysis Program	48
COGO: Run Time Compiler Program	49
COLOGO: Compile, Load and Go Procedure	49
COMPILER: Program to Compile a FORTRAN Program	49
CONPLOT: Aircraft Configuration Plot	49
DAPCA: Development and Production Cost of Aircraft	50
DATCOM: Configuration Design Analysis Program (TRW)	50
DATCOM2: Configuration Design Analysis Program (MDAC)	50
DROPEXS: System File Reduction Procedure	50

CONTENTS (Continued)

Page

~~EDIT: CRT Editing Program......................51~~
ENCYCL: Design Point Performance of Turbojet and Turbofan Engine......................51
ENDODN: End ODIN Procedure......................51
FNT: File Name Table Program..................51
GENENG: A Program for Calculating Design and Off-Design Performance fcr Turbojet and Turbofan Engines........................52
GENENGII: A Program for Calculating Design and Off-Design Performance of Two and Three Spool Turbofans with as Many as Three Nozzles.................................52
GEOMETRY: Body Coordinate Generator................52
GOGET: Modified Data Cell Retrieval Program.....52
HABACP: Hypersonic Arbitrary Body Aerodynamic Computer Program........................53
HEADING: Heading Program..........................53
IMAGE: Configuration Display Program............53
IGM250: Picture Generator for CDC 250 CRT........54
JOBTIME: Simulation Timing Program................54
MAC: Macro-Processor..........................54
MINIVER: Aerodynamic Heating Program..............54
MYPROGRAM: Execute a Compiled Program.............54
ODINEX: A Computer Program for Linking Other Computer Programs.......................55
PADS: Performance Analysis and Design Synthesis Program.......................55
PANEL: A Program for Generating Panelled Configuration Geometry...................55
PLOTSV: Procedure for Saving CALCOMP Plots.......56
PLOTTER: An Independent Plotter Program...........56
PLT250: Independent Plot Program for CDC 250.....56
PLTVIEW: Program for Generating Separation Plots..56
POST: A Program to Optimize Simulated Trajectories............................56
POSTDATA: POST Plot Data Generation Program........57
PRESTO: Program for Rapid Earth-to-Space Trajectory Optimization..................57
PRICE: A Program for Improved Cost Estimation...57
PRINTER: Procedure for Printing Previously Generated Output Information.............58
REPORT: Report Generator........................58
ROUTECC: Data Routing Procedure...................58
ROUTEXP: Dynamic Printing Procedure...............58
RSTART: Restart Procedure.......................59
RTEXT: Data Cell Label Retrieval Program........59

CONTENTS (Continued)

		Page
SBOOM:	Sonic Boom Prediction for Shuttle Type Vehicles	59
SEPARTE:	Program to Simulate Separating Stages of Launch Vehicles	59
SKINF:	Turbulent Skin Friction Drag Program	60
SSAM:	Swept Strip Aeroelastic Model	60
SSSP:	Space Shuttle Synthesis Program	60
TOLAND:	Take-Off and Landing Program	61
TOP:	Trajectory Optimization Program	61
TOPLOT:	Plot Generator for TOP	62
TREND:	Subsonic/Supersonic/Hypersonic Aerodynamic Trade-Off Program	62
VAMP:	Volume, Area and Mass Properties	62
VARIAN:	Data Transfer Procedure	63
VSAC:	Vehicle Synthesis for High Speed Aircraft	63
WAATS:	Weights Analysis for Advanced Transportation Systems	63
WDRAG:	Zero-Lift Wave Drag Program	64
WETTED:	Wetted Area in Reference Length Program	64

APPENDIX B: THE ODIN EXECUTIVE PROGRAM....................65
- Language Structure and Usage....................67
- Construction and Maintenance of Data Bases....................71
- Construction of Simulation Streams....................77
- Retrieval of Design Information....................80
- Report Generation....................83
- Technology Program Interface Requirements....................84
- Excluded Names and Special Options....................86

LIST OF FIGURES

Figure Number		Page
1	The Optimal Design Integration (ODIN) System	4
2	Sample of the Programs in the ODIN Library	7
3	Representative Utility Programs	8
4	Contributions to the ODIN Library	9
5	Confidence Levels of the Technology Modules in the ODIN Library	10
6	The ODIN System at the Langley Research Center	12
7	Control and Communication Language Examples	13
8	Data Base Media	15
9	ODIN System Hierarchy	17
10	Program Execution Control	19
11	ODIN Data Base Management	20
12	Program Data Construction	21
13	Steps in Using ODIN	23
14	Study Variables	26
15	Program Arrangement	27
16	Summary Geometric, Weight and Aerodynamic Characteristics	30
17	Configuration Selected	31
18	Model of Refined Orbiter Configuration in the Langley Unitary Plan Wind Tunnel	32
19	Orbiter Surface Temperature Study	33
20	Results of the Orbiter Surface Temperature Study	35
B-1	Scenario for the Execution of One Technology Program	66
B-2	ODIN Executive Language	
	(A) Control Directives	68
	(B) Communication Commands	69

PAGE BLANK NOT FILMED

ODIN: OPTIMAL DESIGN INTEGRATION SYSTEM

By C. R. Glatt and D. S. Hague
Aerophysics Research Corporation

SUMMARY

The ODIN system is a digital computer program complex for the synthesis and optimization of reusable launch vehicle preliminary designs. The system consists of a library of technology modules in the form of independent computer programs and an executive program, ODINEX, which operates on the technology modules.

The ODIN library contains programs for estimating all major flight vehicle system characteristics, such as geometry, aerodynamics, propulsion, mass properties, trajectory and missions, economics, steady-state aeroelasticity and flutter, and stability and control. In addition, a set of utility programs is available as aids in the linkage of the technology program library and interpretation of simulation results.

The executive program controls the design synthesis by operating on the program library under control of a user specified synthesis procedure established by the input data. Any set of vehicle component matching and sizing loops can be defined. There is no effective limit on the design sequence "topology" which may be employed in a simulation. A data link is provided by a data base which is accessible by all programs through the executive program. The data base interfaces are established in the input stream of the library programs.

The data base used by the ODIN system consists of the dynamically constructed name addressable region of online storage which is accessible to the executive program in segments. Each segment represents a subset of the total data available and is retrieved as required to satisfy data interface requirements. In addition, the executive program can address all or portions of existing structured data sets for insertion at any part in the input stream.

The orbiter wing design study and the orbiter skin temperature study described in this report are representative samples of the type of design support activity which can be accomplished with the ODIN system.

INTRODUCTION

The design of an aerospace vehicle demands the involvement of specialists from all engineering disciplines. These specialists must carefully integrate the requirements of the disciplines in order to obtain the best design for the specified mission spectrum. Many iterations are usually required before a suitable vehicle design emerges. The design iterations usually require from one to three months in the preliminary design phase and can be much longer in the later design phases. Each discipline involved in the design process generally is constrained by the requirements of other disciplines, and much laborious data communication is required at each step. The interface among disciplines is often ill-defined leading to untimely or inaccurat information transfer. Under these circumstances, decisions affecting the usefulness and the schedule of the end product can be based on poor or unreliable information.

The above factors have lead to increased use of the high speed digital computer to expedite the design analysis and improve the quality of the design information. Automation of the individual disciplines has played an increasing role in the design process for more than a decade. Structural analysis and system performance have led the way in large scale computer applications, although nearly every aspect of the design process has been automated to some degree. More recently, the merging of the technologies into a single preliminary design tool has been attempted. One successful preliminary design tool is exemplified by references 1 and 2. Here a complete synthesis of the design and mission analysis is contained in a single computer program.

The confidence gained in early simulation attempts has led to th development of more detailed and complex modules. References 3 through 6 are examples of recently developed simulation tools. However, most modern day integrated design programs tend to suffer from one or more of the following deficiencies:

1. Lack of depth in the analysis techniques.

2. Insufficient or inflexible data intercommunication.

3. Poor response time to rapidly changing design requirements.

4. Excessive computer core requirements.

By-and-large the technical depth is available in independent technology programs. The pattern of development of these programs has been the generalized multiple option approach suitable to the analysis of many classes of vehicles, each class being represnted by input data. The problem arises in combining the technology programs into a design synthesis program. Computer core limitations require that resulting synthesis programs be generally more limited in scope than the individual program. As a result, the synthesis programs tend to become obsolete very quickly as the design process evolves. Very often the obsolescence occurs before any effective use can be made of the synthesis program.

The deficiencies described above led to the development of a design synthesis system called ODIN (Optimal Design Integration). The ODIN system shown schematically in figure 1 is a very large scale computer program complex consisting of a library of independent computer programs, an executive computer program and a data base of design information. The executive program allows the selective use of the library programs as elements of a comprehensive design simulation. The data base provides the common information link among the library programs. The ODIN system was initially developed during 1971 and 1972 under parallel contracts with the National Aeronautics and Space Administration, Langley Research Center (NAS1-10692) and the United States Air Force, Flight Dynamics Laboratory (F33615-71-C-1480). These studies resulted in two similar computer systems (references 7 through 9) having different technology program libraries but using the same executive program. The original system was developed for the CDC 6000 series computer but has since been converted to the Univac 1100 series computer under contract to NASA Johnson Spacecraft Center (NAS9-12829). The system has found considerable usage at NASA in support of shuttle design studies (references 9 through 12). The USAF has applied ODIN to air-to-surface missile studies under contract F33615-73-C-3039. References 13 through 26 exemplify the technology modules which have been used with the ODIN procedure. All the depth of analysis in each technological area is maintained and the computer core requirement is no larger than the largest program element selected.

The technology module program library has been established by an extensive survey of existing computer programs available to the general aerospace industry. Governmental, industrial and academic sources for technology module programs were used in construction of the program library. Programs which form the ODIN library are summarized in Appendix A. The executive program which forms the nucleus of the system is described in Appendix B.

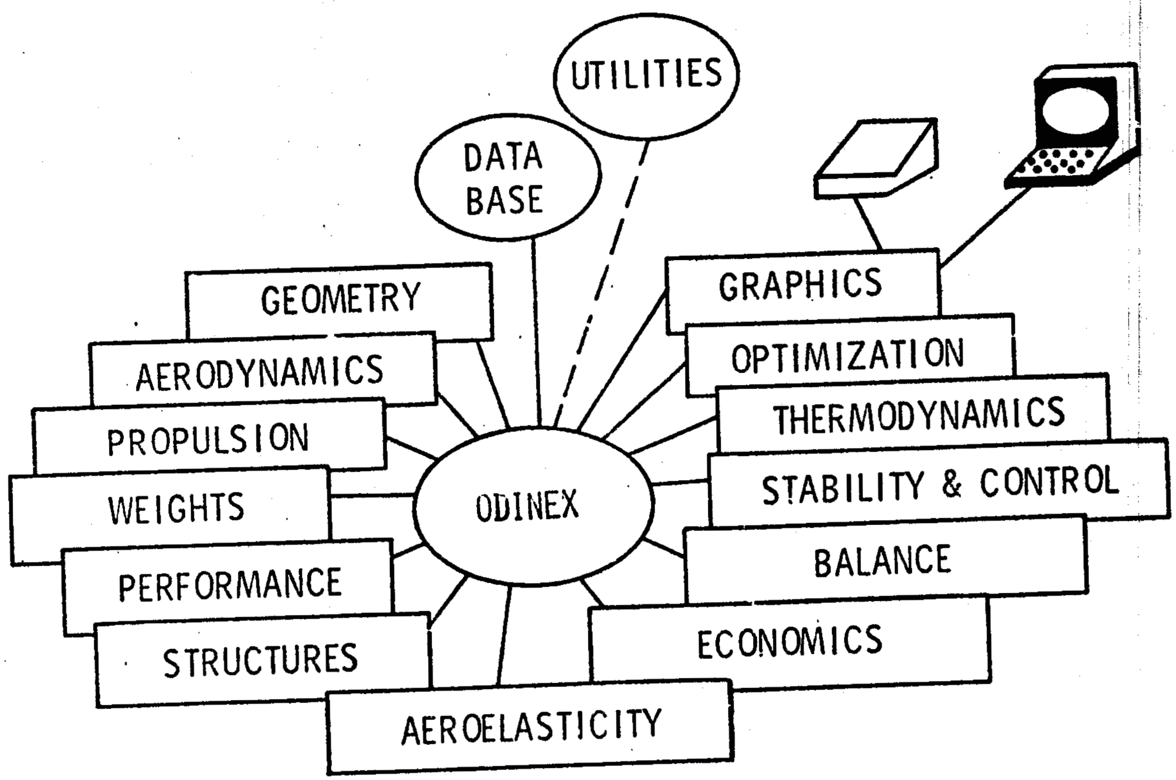


FIGURE 1 THE OPTIMAL DESIGN INTEGRATION (ODIN) SYSTEM.

THE ODIN SYSTEM

The components of the system, shown schematically in figure 1, exist in the form of independent computer programs. These computer ~~programs can be technology~~ programs, service programs or utility programs. All of the major technologies, such as aerodynamics, propulsion and structures, are represented in the library of computer programs. The utility elements include design graphics, plotting and report generation. The technology programs provide the real design analysis which must be performed to obtain evaluation of any vehicle design. The utility elements aid the designer in obtaining a better understanding of simulation results and to provide a means of improving or expediting the design analysis. They also permit the designer to transfer or transform data generated by one program for use by other programs or use by the analyst.

Since the system comprises literally millions of source cards, some precautions have been taken to provide a usable system capable of interpretation by designer, engineer and programmer. The major precaution has been the creation of a system which is truly modular in the sense that it consists of many independent computer programs. Any one of these programs can be revised, extended or replaced without affecting the other program elements of the system in any way. External to the technology programs themselves, a data base of common information is maintained by an executive data management system. Each technology program may draw upon the data or replace it as required. As a consequence of the construction concepts employed, the specialists in any technology area are able to phrase the analysis of the design without regard for the other technologies involved other than the interfaces with the common data base. The common data base attributes are defined by the design staff and can consist of all information which is communicated between elements or communicated from the ODIN system itself to the staff. The data represents a subset of the total amount of information generated by all of the programs within a given ODIN simulation. When combined with the normal input data, it is sufficient to completely define the program under study. When combined with the normal output, it represents the data description of the vehicle performance and mission requirements.

The Program Library

The independent program elements which form a basis for the ODIN system are largely written in FORTRAN, although this is not a system restriction. In fact, independent programs written in any other language can be intermingled during a design synthesis, for example, FORTRAN, COMPASS and COBOL. The system of programs provides the designer with the building block approach to vehicle design. Thus, to provide a given analysis, it is necessary to use only those programs required for the particular analysis. Approximately 60 programs reside in the library. A sample of the available programs is illustrated in figures 2 and 3. They consist primarily of technology programs but include service programs and utility programs as well. Each is identified by a mnemonic which is used when referring to the program during a simulation. The programs generally perform some engineering analysis function such as aerodynamics, performance or structural analysis. For example, the aerodynamics estimation programs provide theoretical and empirical methods spanning most aerospace vehicle shapes in common use over a complete Mach number spectrum. The performance programs provide vehicle performance methods ranging from the simplist segmented mission analysis to the most complex variational optimization techniques. Service programs perform computer aided analysis functions such as optimization, display graphics and report writing. During a simulation, system performance and constraint criteria are generated which are later used for evaluation of the vehicle design (either automatic or manual). Service programs are aids in this evaluation. The utility programs generally transfer information or transform data to the benefit of the user or the simulation and perform such tasks as compiling and executing interface programs. Further, utility functions allow the suppression of unnecessary printed information generated during design simulation and the routing of design reports to alternate sites. Abstracts of many ODIN programs are appended to this report.

Most of the technology programs were in existenace before development of ODIN. Figure 4 illustrates the major contributors to the ODIN library. The contributions are largely a result of NASA or USAF sponsored research studies. There is no limit to the number of programs which can be incorporated into the library or to the number of programs which can be called upon in a given simulation. Integration of new technology programs into the system is a relatively simple task and new programs are constantly being integrated into the system as a need arises.

Figure 5 shows the relative confidence levels of the various technology programs in current design activities. These confidence levels are, however, merely an estimate of the present capability and would have to be revised as new programs are included. For example, in structures there are many programs

GEOMETRY
- AIRFOIL
- GEOMETRY
- HABACP
- PANEL

AERODYNAMICS
- AIRFOIL
- DATCOM
- DATCOM 2
- HABACP
- SKINF
- TREND
- WDRAG

PROPULSION
- ENCYCL
- GENENG

OPTIMIZATION
- AESOP

PERFORMANCE
- ATOP II
- COBRA
- PADS
- POST
- PRESTO
- SSP
- TOP
- VSAC

STABILITY AND CONTROL
- DATCOM
- DATCOM 2
- HABACP
- TREND

THERMODYNAMICS
- HABACP
- MINIVER
- TREND
- ATOP I'

GRAPHICS
- IMAGE
- PLOTTER
- VARIAN

BALANCE
- VAMP

ECONOMICS
- DAPCA
- PRICE

STRUCTURES
- AFSP
- SSAM

WEIGHTS
- WAATS
- PADS
- VAMP
- SSSP

FIGURE 2 SAMPLE OF THE PROGRAMS IN THE ODIN LIBRARY.

PROGRAM ACRONYM	DESCRIPTION
COLOGO	COMPILE, LOAD, AND EXECUTE A FORTRAN PROGRAM
COMPILER	COMPILES A FORTRAN PROGRAM
MYPROGRAM	EXECUTES A COMPILED FORTRAN PROGRAM
PRINTER	PRINTS NORMAL PROGRAM OUTPUT GENERATED IN THE PREVIOUS EXECUTION
PLOTSV	PLOT SAVE PROCEDURE FOR CALCOMP PLOTS
REPORT	GENERATES USER SPECIFIED STATUS REPORT
ROUTECC	ROUTES SELECTED OUTPUT INFORMATION TO THE CENTRAL COMPUTING FACILITY
ROUTEXP	DYNAMICALLY PRINTS REPORT AT ORIGINATING TERMINAL WHILE JOB IS EXECUTING
NEWPROC	ALLOWS EXECUTION OF AN ARBITRARY SEQUENCE OF CONTROL CARDS

FIGURE 3 REPRESENTATIVE UTILITY PROGRAMS.

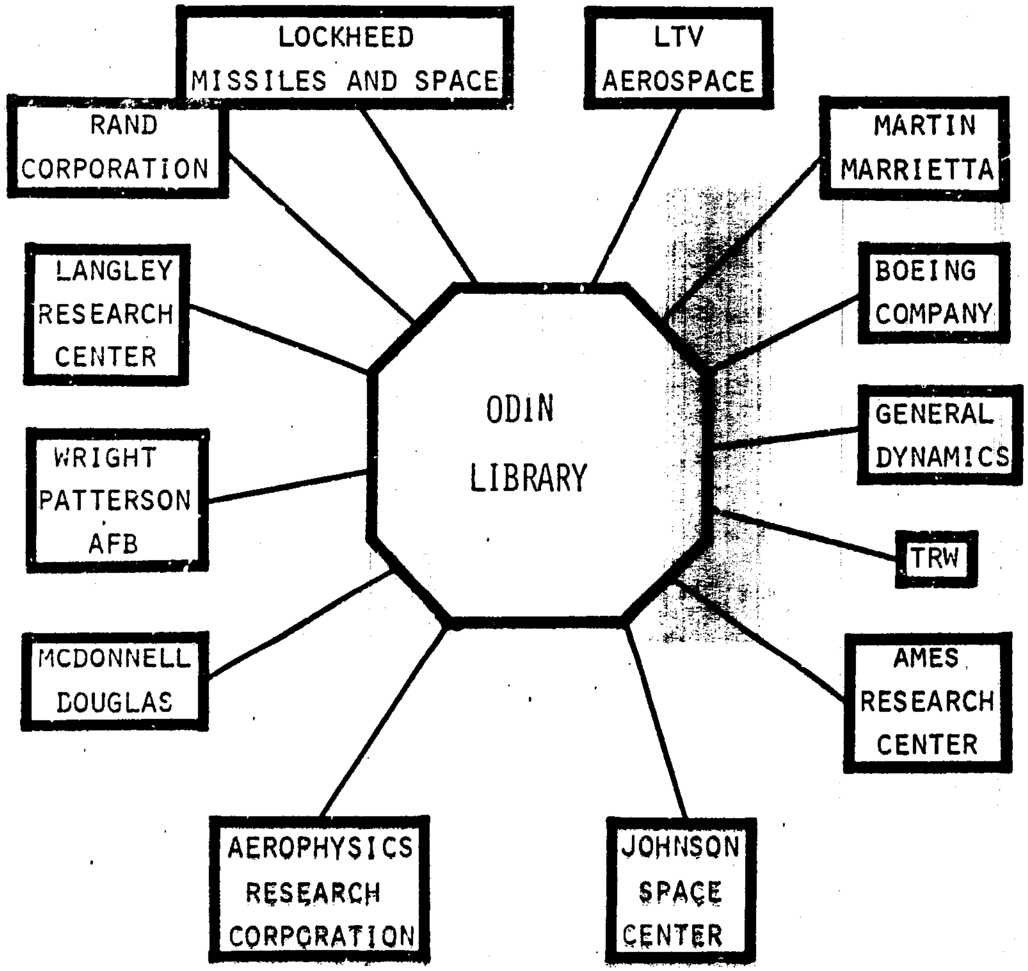


FIGURE 4 CONTRIBUTORS TO THE ODIN LIBRARY.

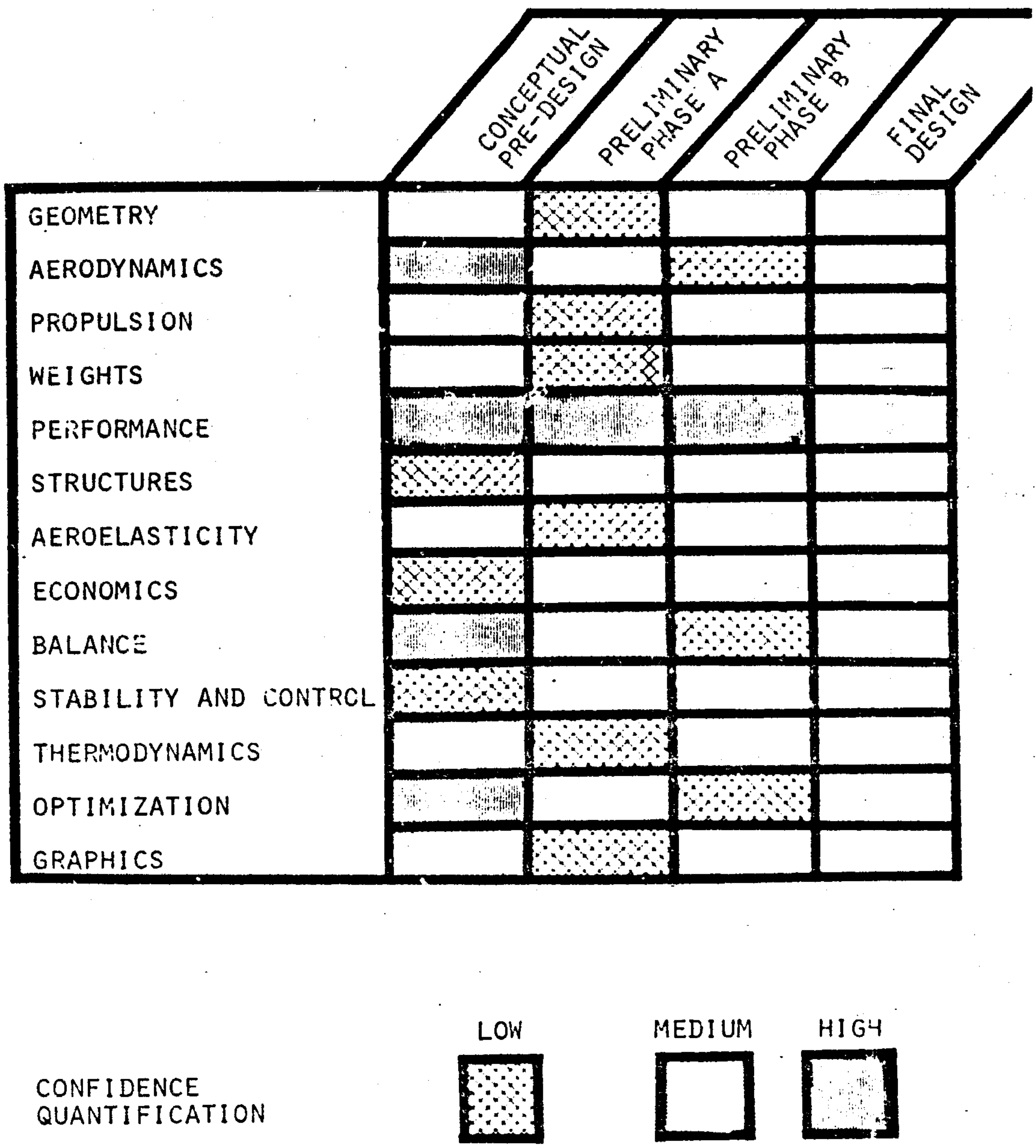


FIGURE 5 CONFIDENCE LEVELS OF THE TECHNOLOGY MODULES IN THE ODIN LIBRARY.

currently available that would enhance the confidence level in this particular technology and when the need arises these programs will be integrated into the system. On the other hand, performance programs for detailed analysis are available and widely used in the ODIN system now.

The ODIN system, as implemented at Langley Research Center, is shown in figure 6. The data cell system is used for permanent storage of the ODIN library, the design data bases and the ODINEX executive system. During execution of a simulation, the disks are used for temporary storage, construction and modification of all files. Tape units may also be used for both temporary and permanent storage as well as storage of information to drive graphical display and milling equipment. While the interactive terminal is not extensively used at this time, its capability has been demonstrated on some problems and its use will grow during the next few years.

The Executive Program

The execution of a sequence of independent programs requires the ability to link control card sequences in the computer. Once this linkage is established, computer programs can be executed in any order by adjusting the order of execution of the control card sequences. In the ODIN system, each control card sequence represents the retrieval and execution of one or more computer programs. The run stream for a given simulation is constructed by the executive system from stored control card sequences based on commands to ODINEX. Representative control and communication commands required to construct a run stream are illustrated in figure 7. The ODIN system usage, therefore, depends not only on the storage of the library programs but also the storage of control card procedures for each program in the library. The executive system constructs and maintains the control card sequences for all of the library programs in a separate data base segment. To use the ODIN system, basic data for each computer program requested must be set up in the usual manner for that program operating independently of the ODIN system. The analyst or team of analysts then defines the sequence of programs to be executed together with the affect of all design parameters on the input of each program. The latter effect is accomplished by placing the appropriate data base interfaces within the input data in accordance with the instructions in Appendix B.

The design parameters are placed in the design data base segment so that they will be available to each program in the design sequence. The simulation then commences using nominal design

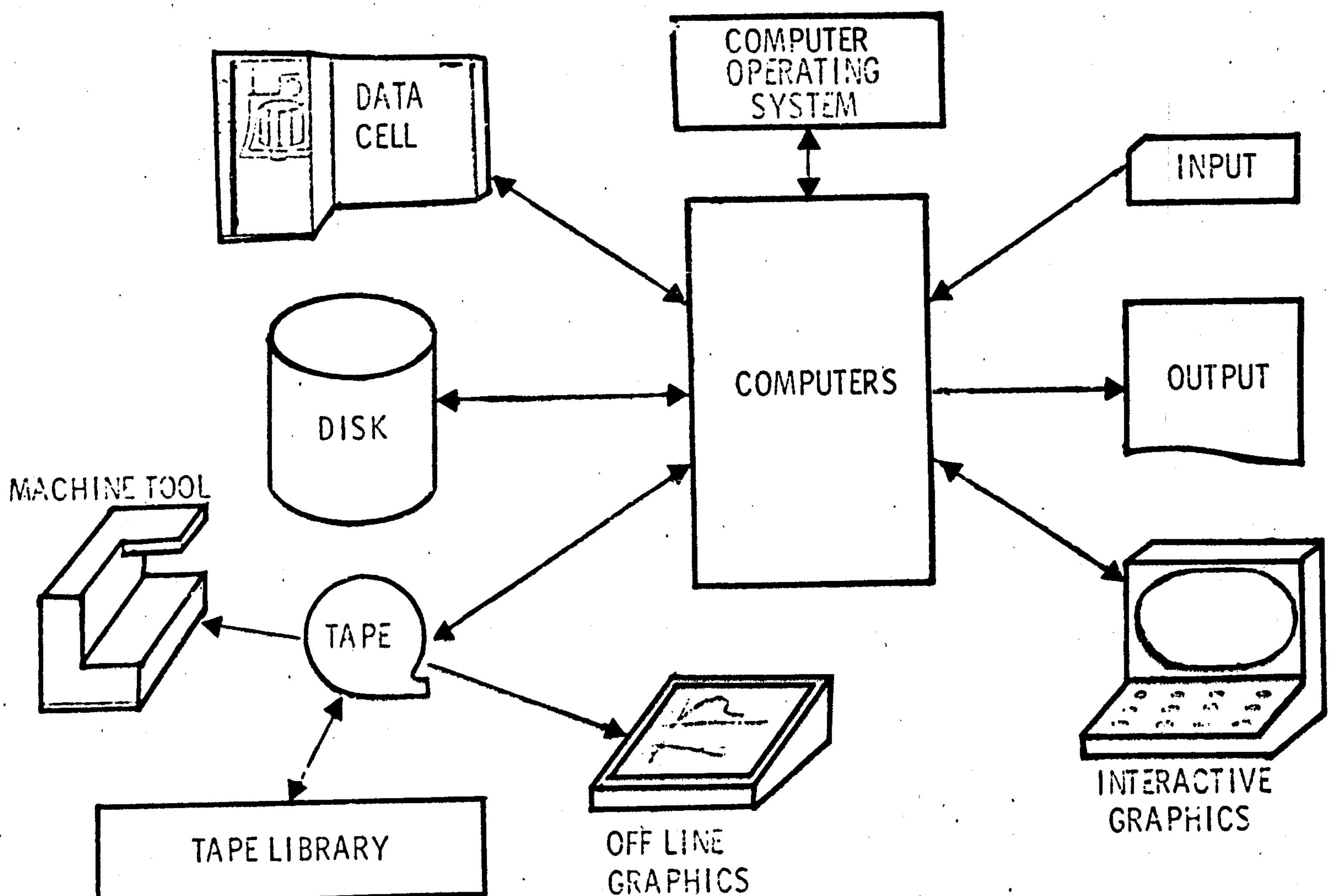


FIGURE 6 The ODIN system at the Langley Research Center

COMMAND	DESCRIPTION
'CREATE NAME'	CREATE A DATA BASE
'ADD NAME = VALUE'	ADD OR ALTER DATA BASE INFORMATION
'EXECUTE NAME'	EXECUTE AN ODIN PROGRAM
'NAME'	RETRIEVE DATA FROM THE DATA BASE
'INSERT NAME'	MERGE INPUT DATA FROM ALTERNATE SOURCES
'LOOP TO NAME'	BRANCH TO AN ALTERNATE POINT IN THE DESIGN SEQUENCE
'DESIGN NAME'	LABEL ALTERNATE POINTS IN THE DESIGN SEQUENCE
'IF NAME , OP , NAME'	CONDITIONAL BRANCHING INSTRUCTION

FIGURE 7 CONTROL AND COMMUNICATION LANGUAGE EXAMPLES.

variables. The executive program interrogates input of each technology program and places requested information from the data base into that input stream. As the simulation proceeds, each program obtains the current value of the design parameters from the data base. A common method of running a simulation is to use a parametric variation or optimization module as the final progra executed in the sequence. The final program collects the relevan system characteristics which are to be evaluated and perturbs the design parameters. The perturbed set of design parameters replac those residing in the data base and some portion of the simulatio sequence is repeated. The repeated sequence defines perturbed system characteristics. The perturbation process is repeated until an optimal design or a parametric analysis is complete. Whereupon, the final results can be plotted or evaluated on a system performance basis. During the simulation, all information required to fully define the problem at the level of analysis requested is stored in the data base. The level of analysis can vary from simulation to simulation and the data base can be defined to suit the amount of information being stored. Upon completion of the simulation, the data base can be interrogated usin the report generation capability within the executive system or the plot capability residing in an independent ODIN computer program to compose a final user oriented description of the vehicle design.

It should be noted that the design data base segment contains only interprogram data and that the flow of all data to and from the data base and the program elements is completely controlled by the executive program. The management of the archival data and one-time input data to the individual programs is not managed by the executive program. When a program is being executed, it is operating on input data which is identical in format to that which it uses when operating as an independent program. This is the key element to the modular structure of the ODIN system. It assures that the analysis function of each program in the library can be examined independently of the other analysis programs. Without this feature, examination of complex interconnections between analysis programs would become extremely complex, and in view of the system size, of doubtful validity.

Data Base

The concept of a data base has been defined in many ways. Some definitions involve computerized storage of information and othe do not. Examples of data base storage media are illustrated in figure 8. A data base may be simply a handbook of information t

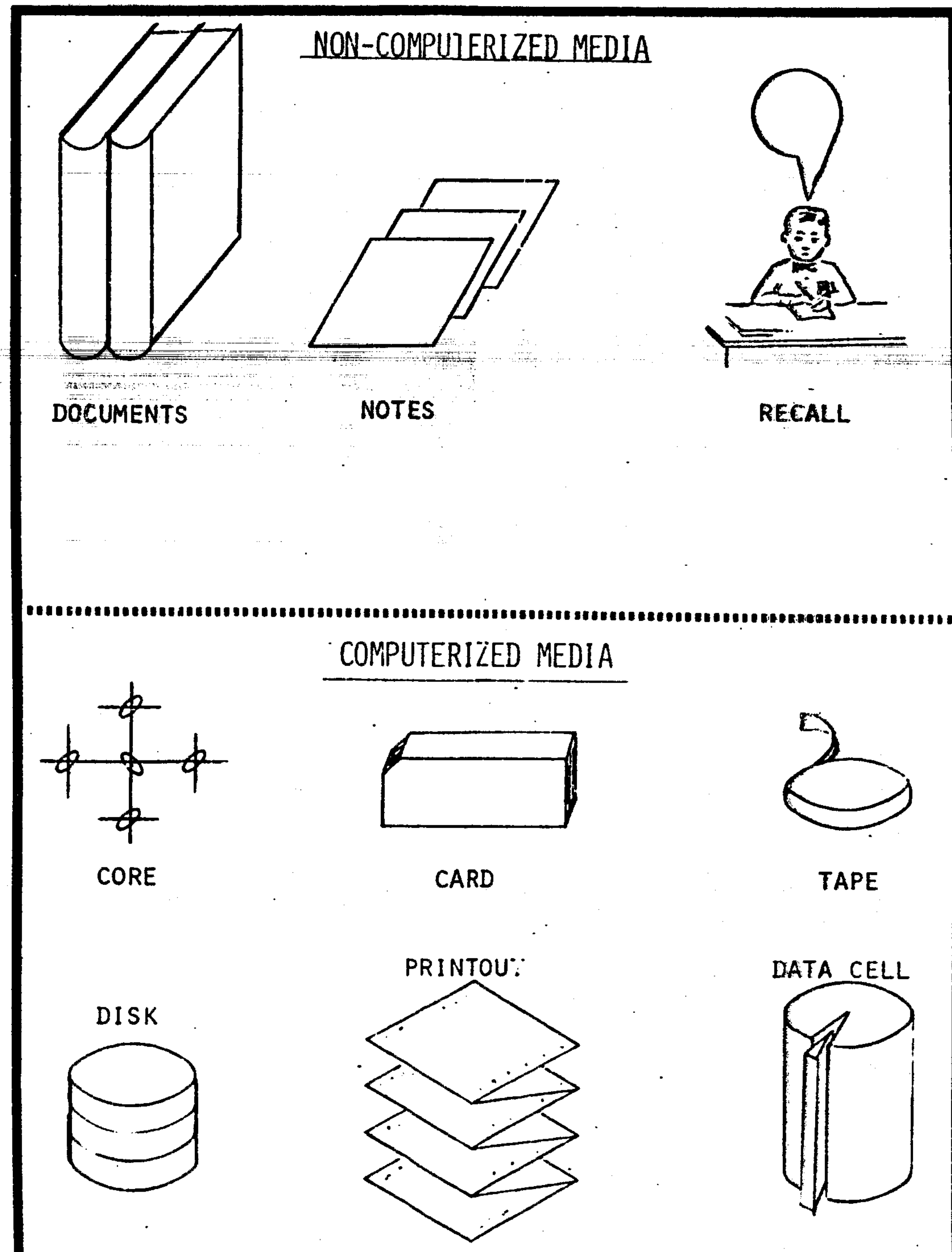


FIGURE 8 DATA BASE MEDIA

which an engineer refers when asked to provide technical information about a particular project. The access to the desired information may be via a formal index or simply by human recall. A data base may also be a computerized storage system. To most programmers, a data base is the region of the computer core set aside for storage of data which is common to all computations in a computer program. In the ODIN system, the dynamic data base segment is a region external to the computer core which is set aside for storage of data which is common to various programs in a given simulation.

Other than the obvious differences between the two classes of data storage (figure 8), it is significant to note that noncomputerized media always have some means of access and retrieval. Without it, the information is of little value. On the other hand, the computerized media illustrated are raw storage devices with no inherent means of indexing. The user of these devices must provide methods of access or retrieval of information in the form of computer programs. The ODINEX executive program provides the access and retrieval methods not only for the design data but the library programs and execution procedures.

The design approach to the ODIN system, recognizing the basic operational philosophy employed by all computer systems, perform all analysis and utility functions at the program file or data set level. All existing programs and the operating system itself are considered resources to the ODIN system. This design approach has led to the development of a very flexible and easily expansible data management system. Figure 9 shows the hierarchy of the various elements of the ODIN system.

The base of the pyramid represents the storage media. All data sets are stored on one of the devices illustrated. The flow of data to and from the storage media is controlled by the operating system. The data sets are represented by the next level which includes the analysis programs, the utility programs and the data files (input, output, etc.). These programs and data sets are controlled by the ODINEX executive system via input commands from the user. Many of the ODIN programs are data management programs in themselves but ODINEX has two builtin data management functions. It controls the sequence of execution of the individual programs and communicates common information among the programs in the sequence.

One key data management feature of ODINEX is the control of the sequence of execution of individual programs. For this function,

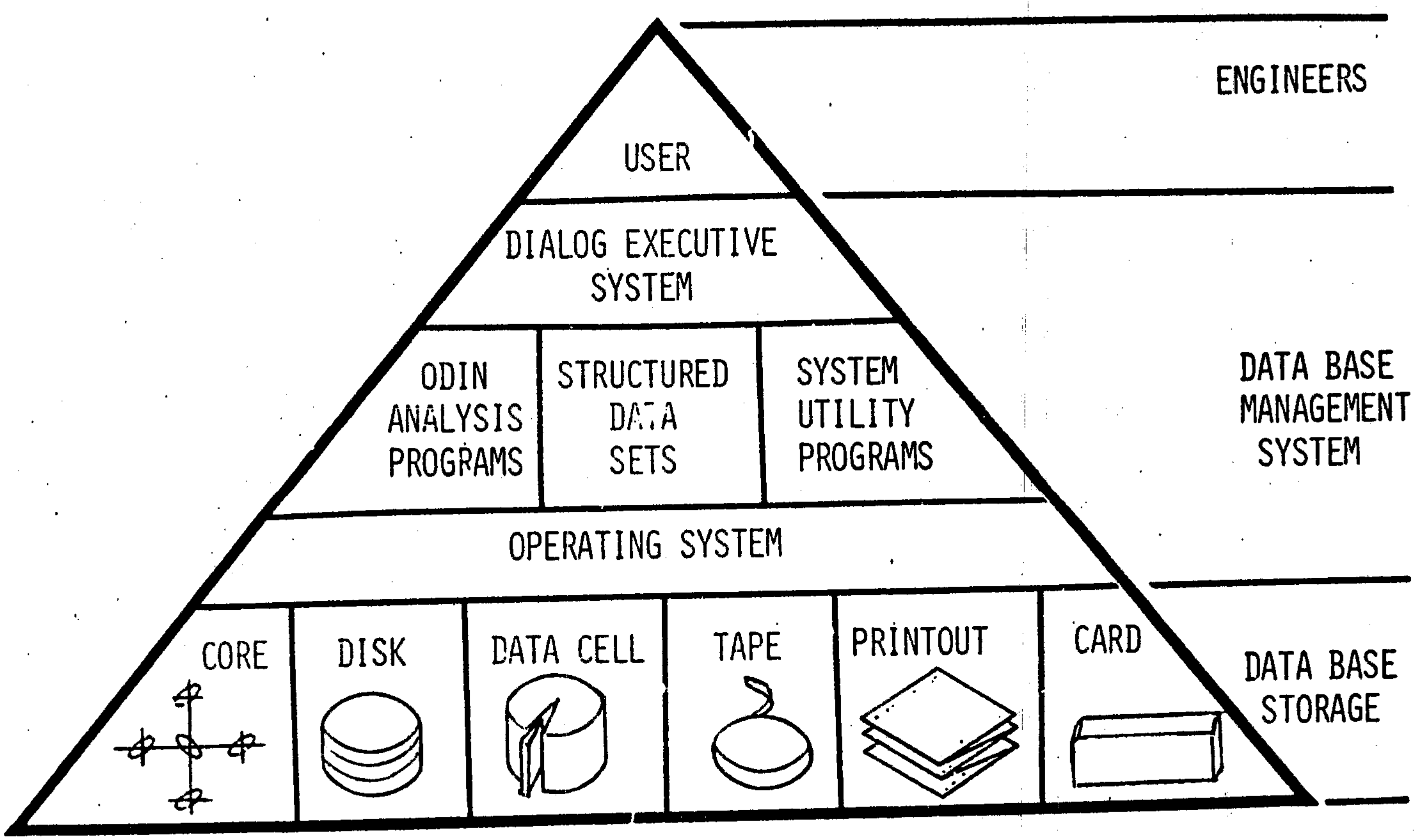


FIGURE 9 ODIN SYSTEM HIERARCHY

ODINEX maintains a data base (or data set) of control procedures for executing the individual programs. Each entry in the data base represents a control procedure for one or more of the programs. These procedures are established once and usually not changed. But as the program library grows, so does the control procedure data base. The control procedure data is itself a data set accessed by ODINEX.

Once accessed, ODINEX constructs an execution procedure sequence from information stored in the data base. This construction is illustrated in figure 10. Included in the execution procedures are instructions to the operating system for accessing the necessary programs and data modules from the library. The operating system then executes the indicated procedures. The ODINEX executive operates independent of, but provides instructions to, the operating system.

The second data management function of ODINEX is to communicate data base information to independent programs. This function is illustrated in figure 11. Here, the storage media is represented on the left. They are controlled by the operating system. The ODIN data base is a collection of information representing a relatively small portion of the entire data requirements. The ODIN data base is constructed dynamically by the ODINEX executive as the execution proceeds. Each program module receives data from the data base and places data into the data base. Each program may also communicate data directly to and from the storage media. Further, data may be taken directly from the user. There is no constraint placed on the data source for any program. When the program is finally executed, the data set which the program "sees" is identical in format to that it would "see" when executed in an independent execution. The ODINEX executive, exercising control over the program module, the data base and the operating system prepare the input for every program specified by the user. The user, exercising control over ODINEX and the human input, directs the analysis process.

The manner in which data sets are created by ODINEX is illustrated in figure 12. The user provides skeletonized input data sets each representing incomplete information for the program sequence but together containing all the essential data for successful execution of the program. The input data sets segments are read by ODINEX and merged into a complete data set. The missing data elements are extracted from the various ODIN data bases. The complete data set is then passed to the applications program for analysis.

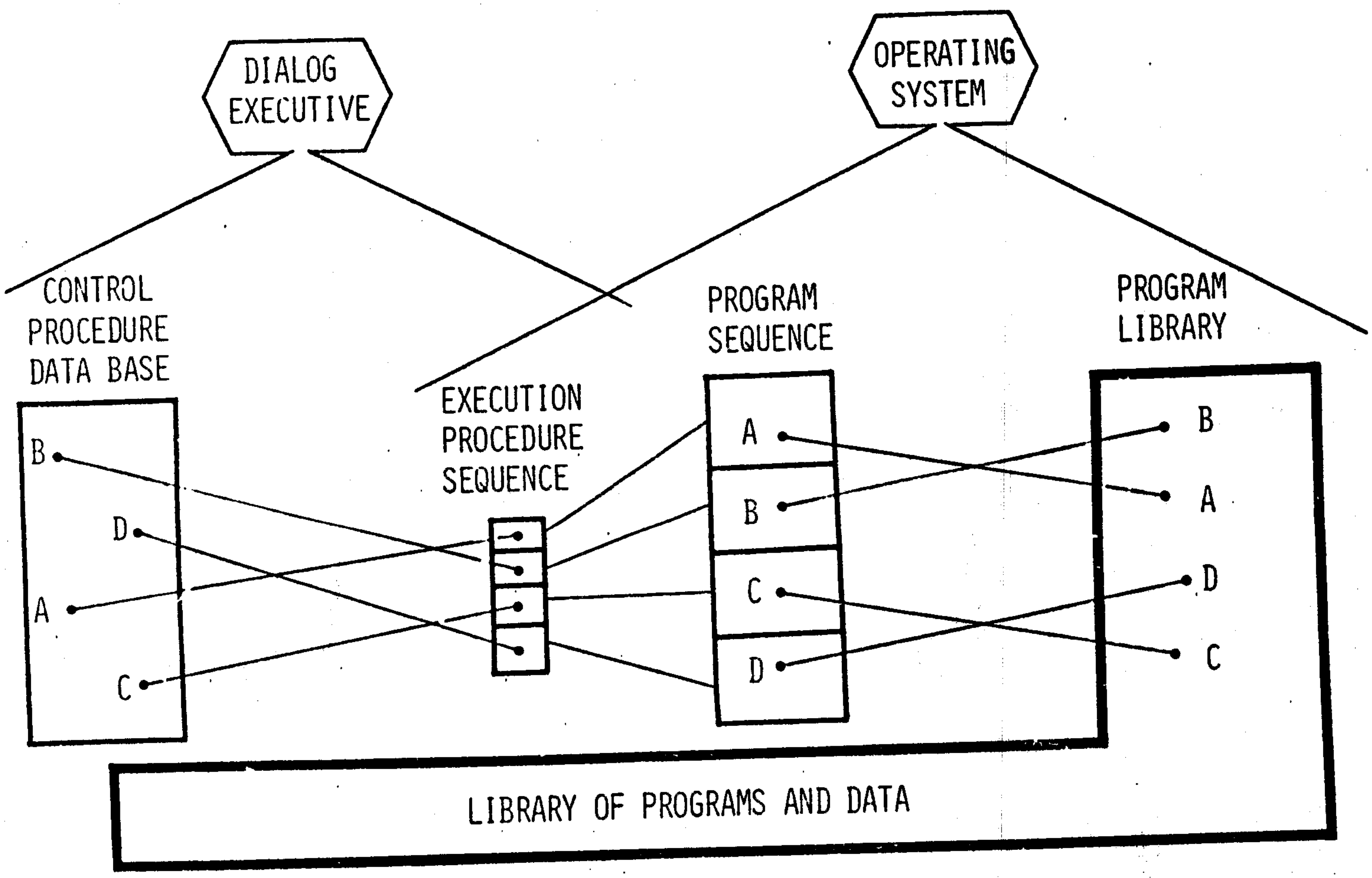


FIGURE 10 PROGRAM EXECUTION CONTROL

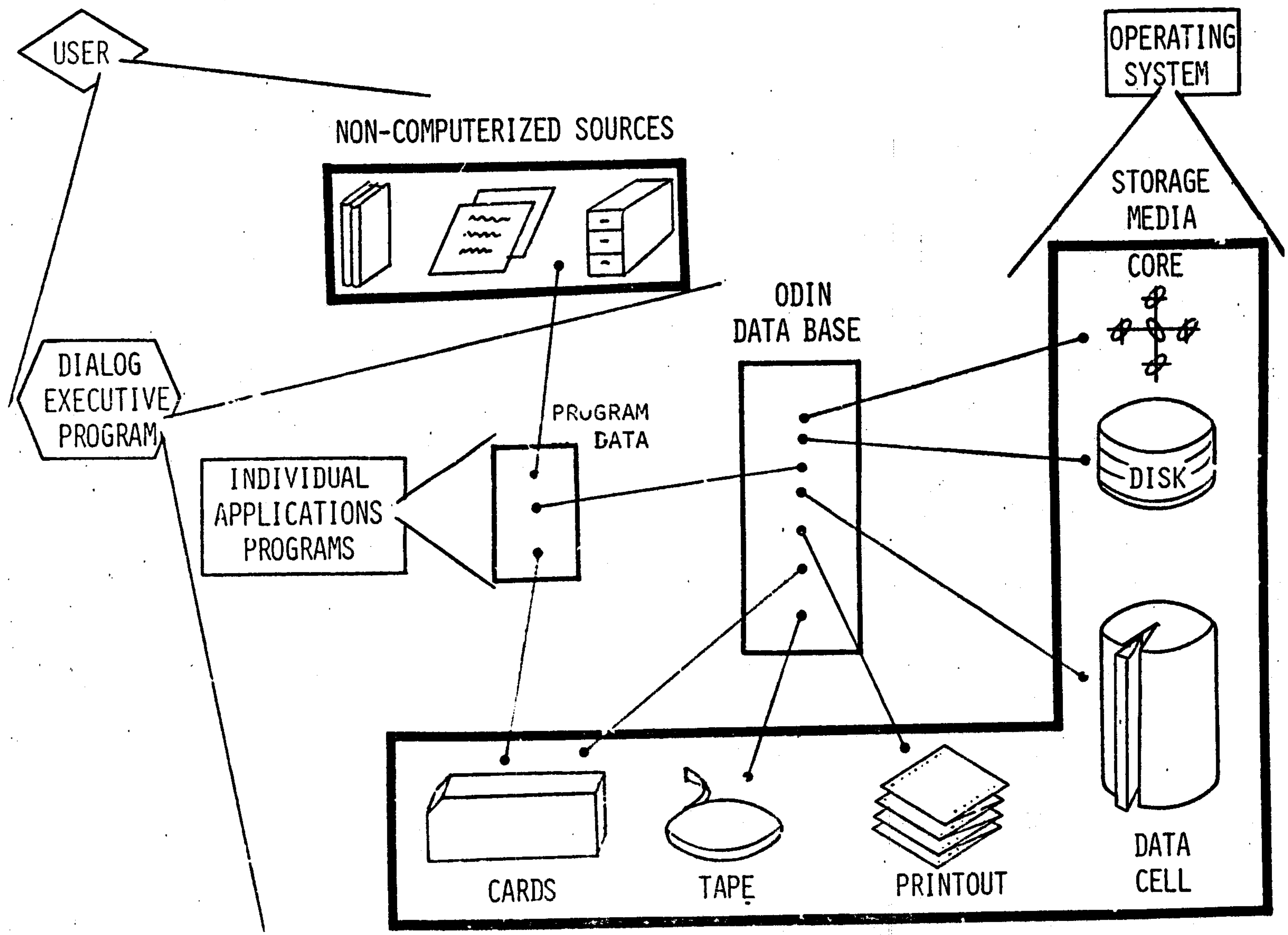


FIGURE 11 ODIN DATA BASE MANAGEMENT

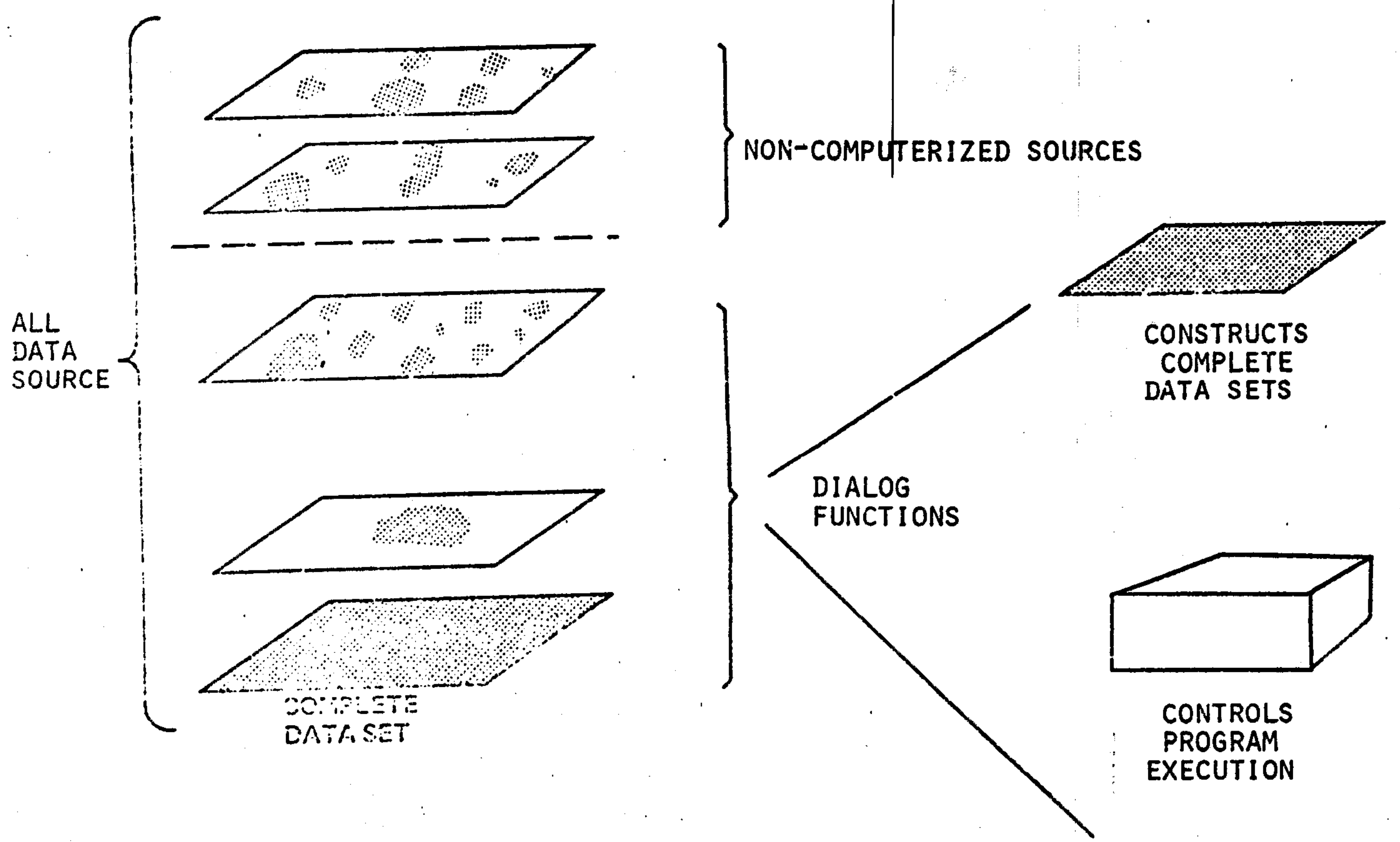


FIGURE 12 PROGRAM DATA CONSTRUCTION

APPLICATIONS

The use of the ODIN system is patterned after normal design practices. Four basic steps, illustrated in figure 13, are required in establishing an ODIN design simulation. First the design tasks must be defined. The technology areas to be considered, together with the depth of analysis to be undertaken, must be established. The second task will be the selection of the technology programs and the sequence which they will be used in the simulation. Program selection will have a direct bearing on the computer resources which will be required. The selection of programs require some understanding of the basic input data f each program and the information each program generates. The task, therefore, may require a team effort if the simulation is large or if many programs are involved. A survey of existing ODIN library programs will aid in the selection process. If a suitable set of programs is not available, one of the following procedures may be employed:

1. Locate the necessary programs from an outside source o

2. Develop the necessary programs to serve the purpose.

The third task is a definition of the interprogram data which will be stored in the data base. Included in this definition a the study parameters, those elements of data which drive the study either through a parametric variation or optimization process. Additionally, the performance criteria must be defined. The performance criteria is the desired study output informatio such as weight or cost of the system under study.

Often there are study constraints, those conditions which must be violated in an acceptable solution. The study constraints i a performance criteria are the basic information used in guidin the selection of values for the study parameter. Interprogram data must also be defined in the data base. These data include intermediate results produced by one ODIN program which are use by other ODIN programs. Finally, the data base may include inf mation which will be used for monitoring the study. The entire data subset, which is stored in the dynamic portion of the data base, is a small subset of the total data manipulated by the various technology programs. However, the data does represent all significant drivers of the simulation tasks and includes th pertinent information normally used for configuration selection The dynamic portion of the data base does not generally include

1. DEFINE THE SIMULATION TASK

   TECHNOLOGY AREAS TO BE CONSIDERED
   DEPTH OF ANALYSIS

2. SELECT THE APPLICATIONS PROGRAM SEQUENCE

   SURVEY THE AVAILABLE ODIN PROGRAMS
   UPDATE THE ODIN LIBRARY
   DEFINE THE EXECUTION SEQUENCE

3. DEFINE THE INTER PROGRAM DATA

   STUDY PARAMETERS
   PERFORMANCE CRITERIA
   STUDY CONSTRAINTS
   ODIN PROGRAM DATA BASE OUTPUT
   MONITORING INFORMATION

4. DECK SET-UP

   SET-UP NORMAL DATA FOR ALL PROGRAMS
   INSERT CONTROL DIRECTIVES
   INSERT COMMUNICATION COMMANDS

FIGURE 13 STEPS IN USING ODIN.

that data representing the detailed geometric aspects of the vehicle, the detailed aerodynamic influence coefficients or the detailed structural criteria. These types of data, if communicated from one program to another, are normally handled by files specifically structured for that purpose. The fourth and final task in setting up an ODIN simulation is the actual deck setup. Deck setup is handled in two phases. In the first phase, the individual program decks are setup in the same manner as they would be if they were to be executed independently. Program options are selected to perform the analysis which will be required by the simulation and a test case is run which represents the nominal vehicle analysis. After all individual program data decks are setup, the control and communication commands are inserted into the decks. This last task effectively creates the overall design sequence.

Since ODIN was installed at Langley, it has been used for design problems of varying degrees of depth of analysis. The applications have been primarily related to the space shuttle and advanced aerospace vehicle concepts. The work on advanced aerosp vehicle concepts has utilized the system the most extensively t date. However, this work will not be discussed since the studi have not been completed. Instead, two completed studies relate to the space shuttle are discussed.

Orbiter Wing Design Study

As the space shuttle program has matured, significant effort ha been devoted to reductions in system weight resulting, in turn, a smaller orbiter vehicle. The payload weight and volume requi ments remained fixed, however, and the variations in potential payload centers of gravity exert an increased influence on the flight characteristics of the smaller vehicle. In addition to wide center of gravity excursions due to the diverse combinatic of payloads, other design requirements such as a maximum allowa landing speed, acceptable unaugmented low speed flying quantiti and stable hypersonic trim at high angles of attack present a formidable challenge to orbiter design engineers.

In the wing design study, an existing orbiter design with known weight characteristics, but unacceptable aerodynamic performanc served as a baseline and the body, tail and internal arrangemer were held constant. The objective was to determine a wing configuration which met the system requirements with a minimum weight. At the conlusion of the study, aerodynamic characteristics of the analytically derived configuration were verified by experimental evaluation at subsonic and hypersonic speeds.

An existing orbiter design, designated the 040A (reference 27) was used as a baseline configuration. The ODIN system was used to determine a wing geometry and location to meet the system requirements in the longitudinal mode. Use of the system allowed the perturbation of the orbiter wing geometry and the evaluation of the weight, balance aerodynamics and stability and control from a selected group of analytical programs within the ODIN library. The specific programs chosen provided pertinent information representing the principal technology areas involved.

The guidelines established for the wing design study were in accord with those outlined and/or implied by the general vehicle requirements of the space shuttle program. The design criteria included a minimum design speed of 150 knots or less at subsonic speeds for a design payload of 18144 kg (40,000 Lb.) located at the half-length station of the payload bay. Minimum design speed ($V_{min,\ des}$) is used herein to denote the minimum level flying speed at $\alpha = 17^{o}$ and sea level standard day conditions for an orbiter having the design payload loading. Additional design criteria included stable subsonic static margin and high angle of attack trim capability ($\alpha_{max} = 50^{o}$) hypersonically over the center of gravity range dictated by the payload envelope. Parameters descriptive of these criteria were recorded in summary reports for various candidate wing designs to enable the user to determine the wing having the most desirable characteristics.

The study variables are shown in figure 14. The primary variables in this study were the local chords and span of the exposed wing. These parameters were varied by using X and Y scale factors (XSF and YSF) to describe variations in the exposed planform of a study wing (i.e. wing planform having XSF = 0.8 and YSF = 1.1 has exposed root and tip chords equal to 80% of the exposed root and tip chords of the baseline wing and an exposed span equal to 110% of the baseline exposed wing span). In this way aspect ratio, leading edge sweep and reference area, were all varied while the trailing edge sweep angle was fixed ($\Lambda_{t.e.} = 0^{o}$) and the taper ratio of the exposed wing was held constant for a major portion of the study. To meet the subsonic static margin requirement, the longitudinal wing position X_{wing} was varied. For some of the wings considered in this study, the trailing edge sweep angle and the elevon area were also varied.

Definition of the execution sequence is shown in figure 15, which illustrates the general arrangement of the technology programs used in the study (see Appendix A). Following initialization, the

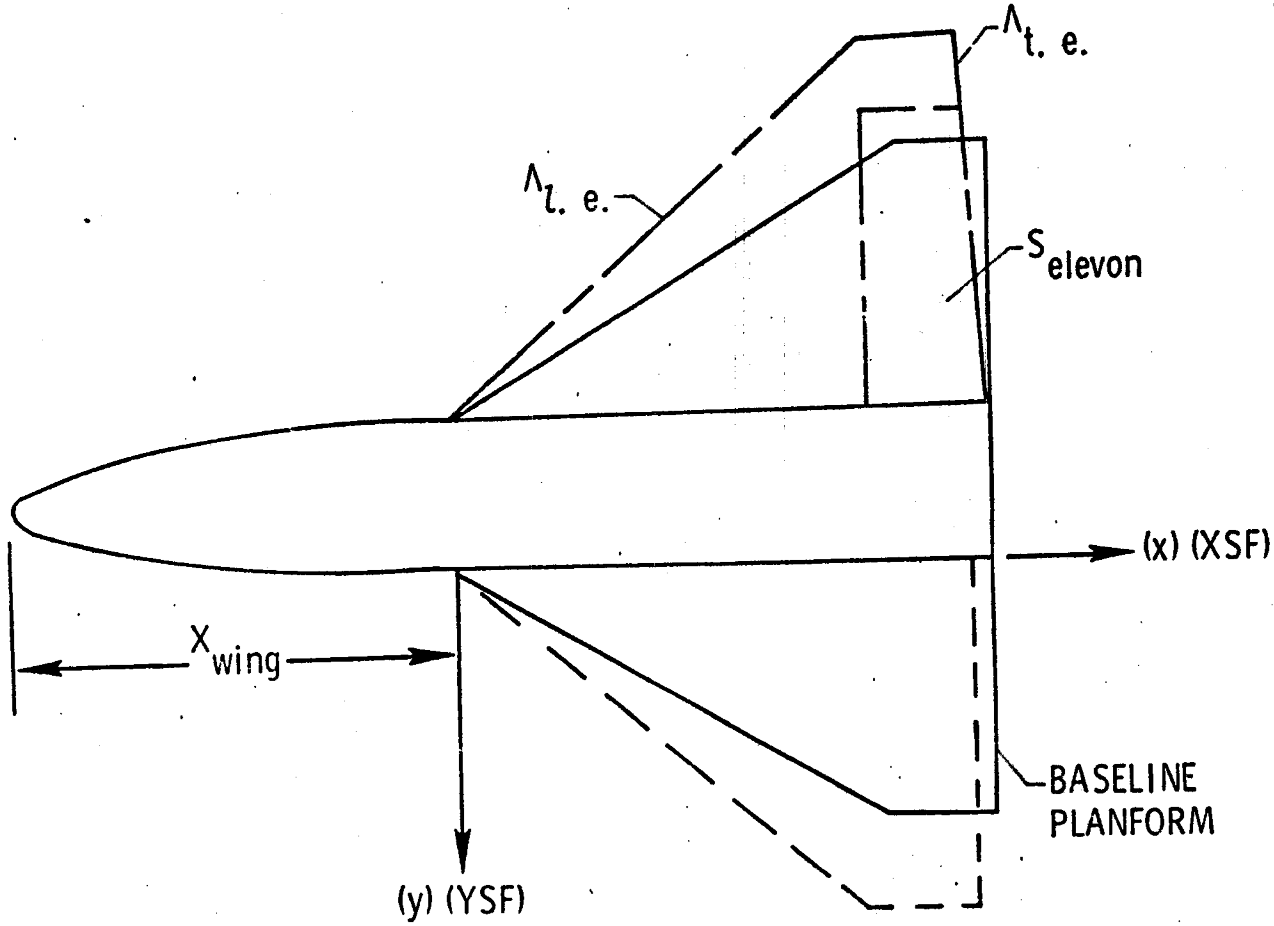
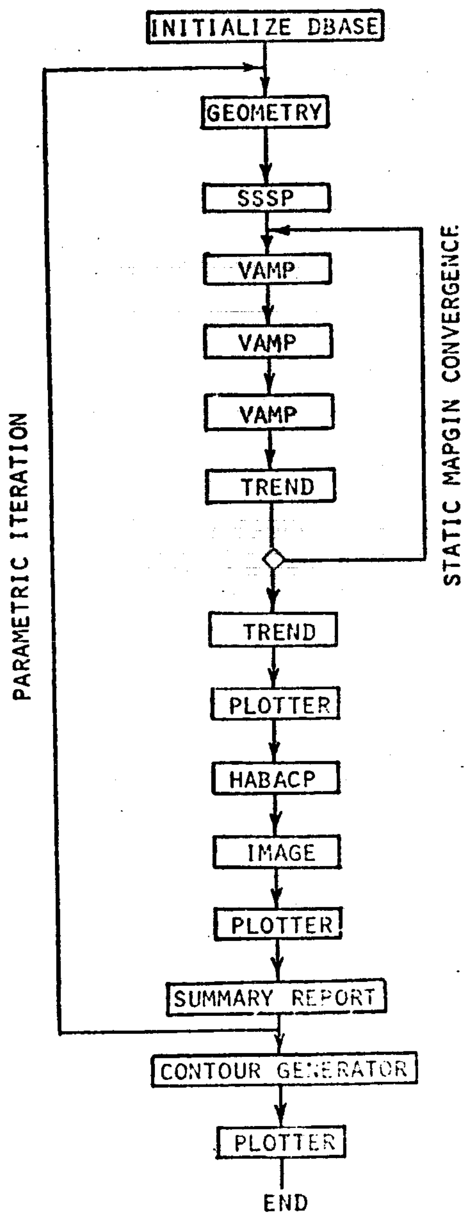


FIGURE 14 STUDY VARIABLES.



- SET STARTING VALUES OF PARAMETERS IN DATA BASE.
- COMPUTE MISCELLANEOUS WING GEOMETRY FROM PARAMETERS.
- STRUCTURAL WING WEIGHT.
- THERMAL PROTECTION SYSTEM WEIGHT.
- LANDING WEIGHT AND C.G. (PAYLOAD OUT).
- LANDING WEIGHT AND C.G. (PAYLOAD IN).
- SUBSONIC LIFT AND DRAG.
- STATIC MARGIN CHECK.
- SUBSONIC STABILITY.
- SUBSONIC LIFT CURVE AND DRAG POLAR PLOTS.
- HYPERSONIC AERODYNAMICS AND STABILITY AND CONTROL.
- THREE-VIEW DRAWING.
- HYPERSONIC NORMAL FORCE AND PITCHING MOMENT CURVES.
- (LANDING SPEED CALCULATION INCLUDED).
- UPDATE CONTOUR MESH IN DATA BASE FOR SELECTED CRITERIA.
- PLOT CONTOURS GENERATED IN LOOP.

FIGURE 15 PROGRAM ARRANGEMENT.

geometry program calculates the detailed geometric characteristics from gross wing parameters. The calculations then proceed sequentially through the technology modules illustrated based upon the calculated wing geometry.

Weight of the body and vertical tail were assumed to be invariant. The wing weight was determined from an aerodynamic surface weight module within the SSSP computer program. This module bases wing weight on historical weight of similar wings having the same structural span, load factor, etc. The thermal protection system weight was determined from the unit weight and the area of the protected surface as determined by VAMP. The VAMP program was also used to determine the centers of gravity of the vehicle for the payload in and payload out condition.

The static margins and trimmed C_L were then obtained from the subsonic aerodynamics program called TREND. Static margins were obtained for payload out and the design payload conditions. The payload out static margin was weighed against a target static margin of $.03\ \bar{c} + .002$, which assured longitudinal stability at the guidelined subsonic flight conditions. If this condition was not met, the system adjusted the longitudinal position of the wing and performed an iterative loop back through the geometry and subsonic aerodynamics program to convergence. After the final subsonic static margin calculation, and the subsonic trimmed lift calculation were completed, the hypersonic characteristics were calculated using the methods programmed in HABACP. The graphics program was then used to illustrate the vehicle panel arrangement and to plot the subsonic and hypersonic aerodynamic characteristics. A summary report provided the pertinent information such as wing geometry, the weight of the vehicle, the center of gravity locations, the minimum design speed and the maximum hypersonic trim angle of attack.

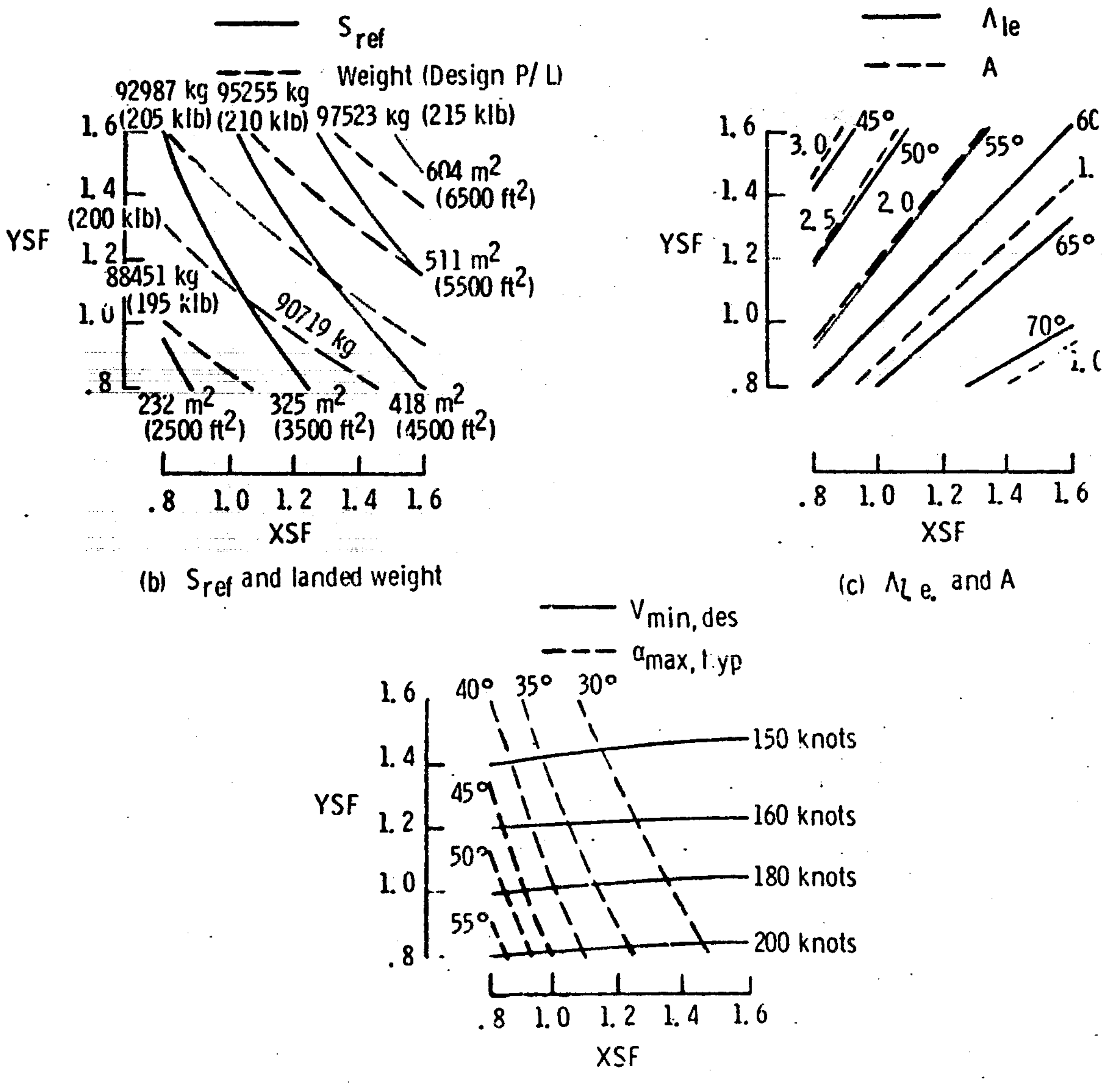
A single pass through the above described simulation generated the characteristics of a single wing design. Initially twenty-five different wing planforms were considered in the matrix of parameters which covered a broad spectrum of possible wing designs. This was achieved by perturbing the ratios of span and chord of the study wings to the baseline and initiating a new cycle. Approximately one hour of computer time was expended in the analytical study of the initial matrix of twenty-five wings. The estimated time saved by using the ODIN system as contrasted to running the individual programs independently to perform a similar study was approximately one-half man-year.

The design and performance data were stored for retrieval in computer generated summary plots, thus enabling the user to isolate the effects of the various design variables. The results shown in figure 16 are presented as constant lines of vehicle landed weight, wing area, aspect ratio, leading-edge sweep, minimum design speed and maximum hypersonic trim angle of attack as functions of the X and Y scale factors which define the various wing planforms included in the study. Examination of the resulting trends indicated that a wing planform having an aspect ratio of about 2.8 and a leading-edge sweep angle of about 45° would provide a minimum weight configuration and satisfy the design criteria.

Using the results of figure 16, some additional wings which had negatively swept trailing edge were considered which also helped the subsonic/hypersonic compatibility problem. The configuration that was finally selected is shown in figure 17. The wing had a 46.8° leading-edge sweep angle and a trailing-edge sweep angle of -11°. The leading-edge sweep angle was slightly larger than 45° because of entry heating concerns. The results indicated that the selected configuration met all of the design requirements with the exception of the maximum trim angle of attack hypersonically wherein the design exhibited a maximum trimmed angle of attack 4° less than the required value. This deficiency could be eliminated by some fuselage nose reshaping (not considered in the wing study) to provide a positive increment in pitching moment. The selected configuration was subsequently tested in the wind tunnel at both subsonic and hypersonic speeds. The analytical aerodynamic estimates obtained were in good agreement with the experimental results. The selected configuration required only minor refinements based on the experimental results to meet the design criteria. The primary refinement was the addition of a small planform fillet to increase lift coefficients at landing attitudes and at the same time provide linearization of the pitching moment curves. A photograph of the orbiter configuration that evolved from this study is shown in figure 18. All of this work is presented in reference 12.

Orbiter Landing Skin Temperature Study

An example of a relatively small problem solved by using the ODIN system is shown in figure 19. A study was required to determine if landing performance or stability and control might be affected by the presence of excessive skin temperatures on the Space Shuttle orbiter. This problem was formulated to determine skin temperature time histories using various technology programs. The MINIVER (reference 21) and ABLATOR (reference 23) programs were quickly integrated into the system. MINIVER was used to obtain convective heat rates along the entry trajectory, and ABLATOR enabled calculations of the associated skin temperature variations



(b) S_{ref} and landed weight

(c) Λ_{LE} and A

(d) $V_{min,des}$ and $\alpha_{max,hyp}$

FIGURE 16 SUMMARY GEOMETRIC, WEIGHT AND AERODYNAMIC CHARACTERISTICS.

Λ_{le} = 46.8°

Λ_{te} = -11.2°

S_{ref} = 315 m^2 (3387 ft^2)

S_{elevon} = 63.1 m^2 (679 ft^2)

λ = .135

A = 2.4

i_{wing} = 1.5°

Payload out:

Landed weight = 72398 kg (159609 lb)

x_{cg} = 0.671l

$C_{m_{C_L}}$ = -0.028 $\bar{c}$

Design payload:

Landed weight = 90542 kg (199609 lb)

x_{cg} = 0.650l

$C_{m_{C_L}}$ = -0.080 c

$V_{min,des}$ (α = 17°) = 150 knots

$\alpha_{max,trim}$ at hypersonic speeds = 46°

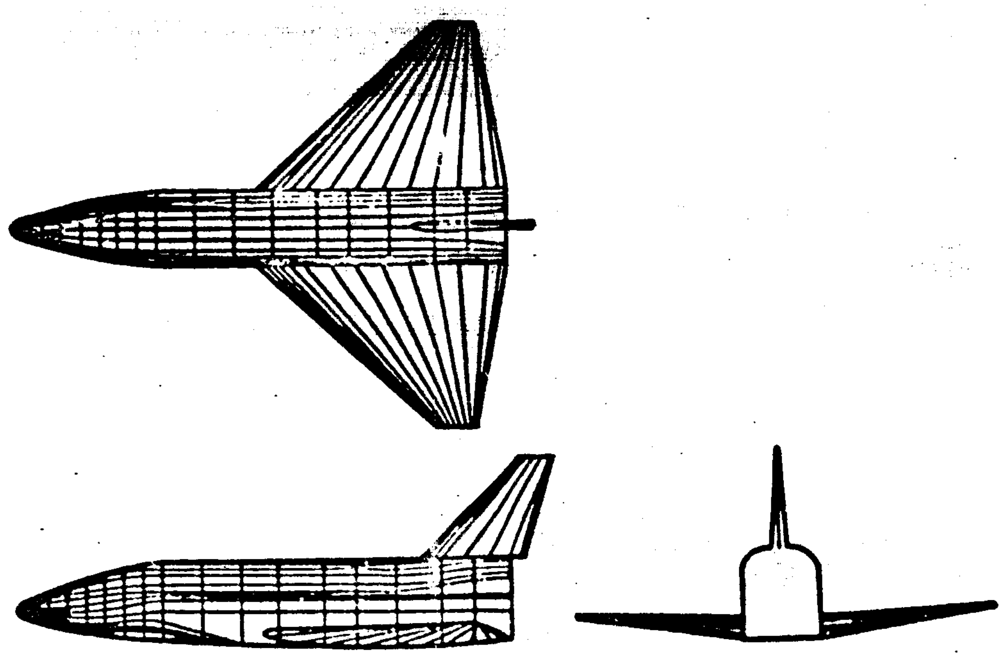


FIGURE 17 CONFIGURATION SELECTED.

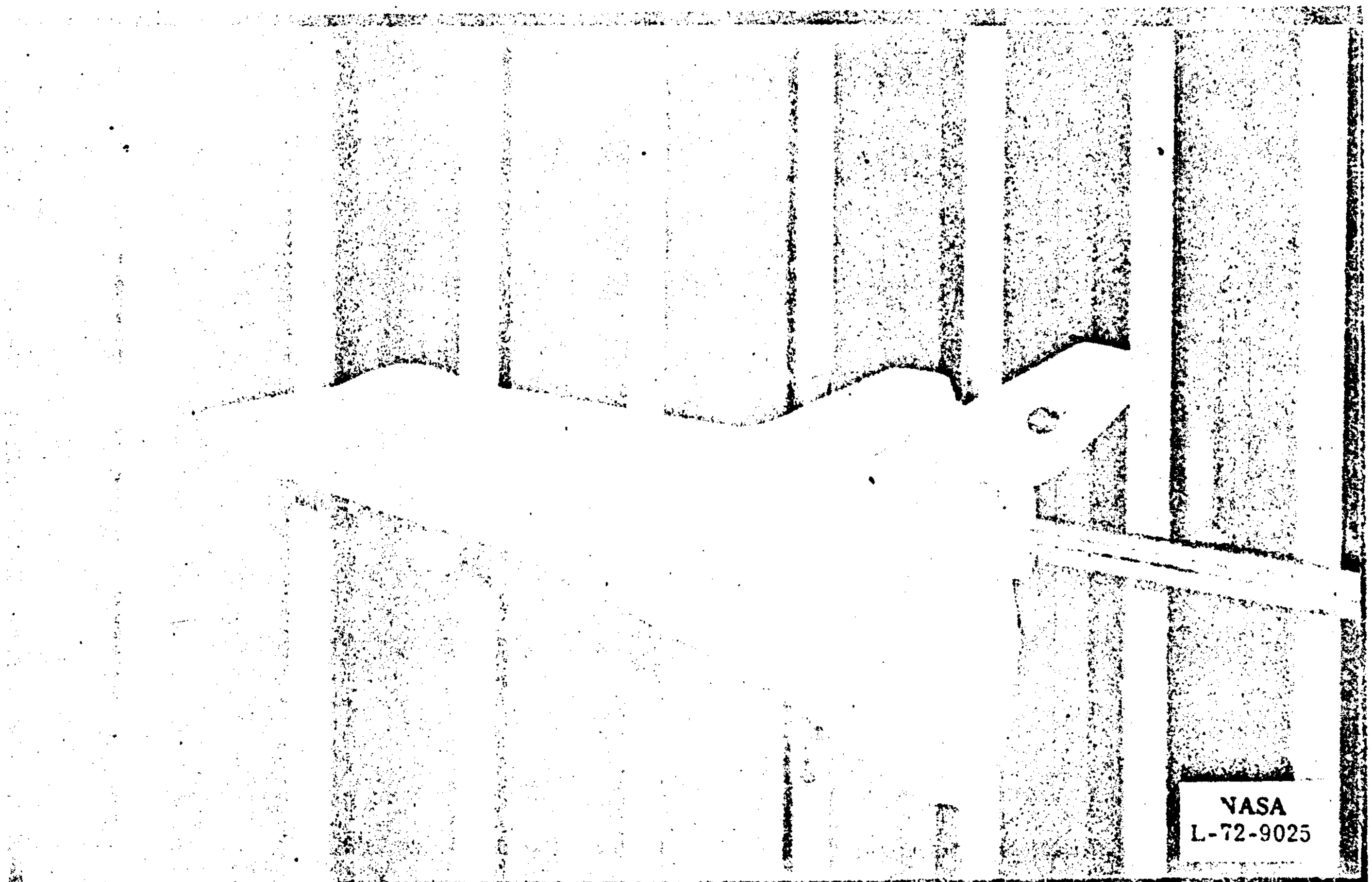


FIGURE 18 MODEL OF REFINED ORBITER CONFIGURATION
IN THE LANGLEY UNITARY PLAN WIND TUNNEL.

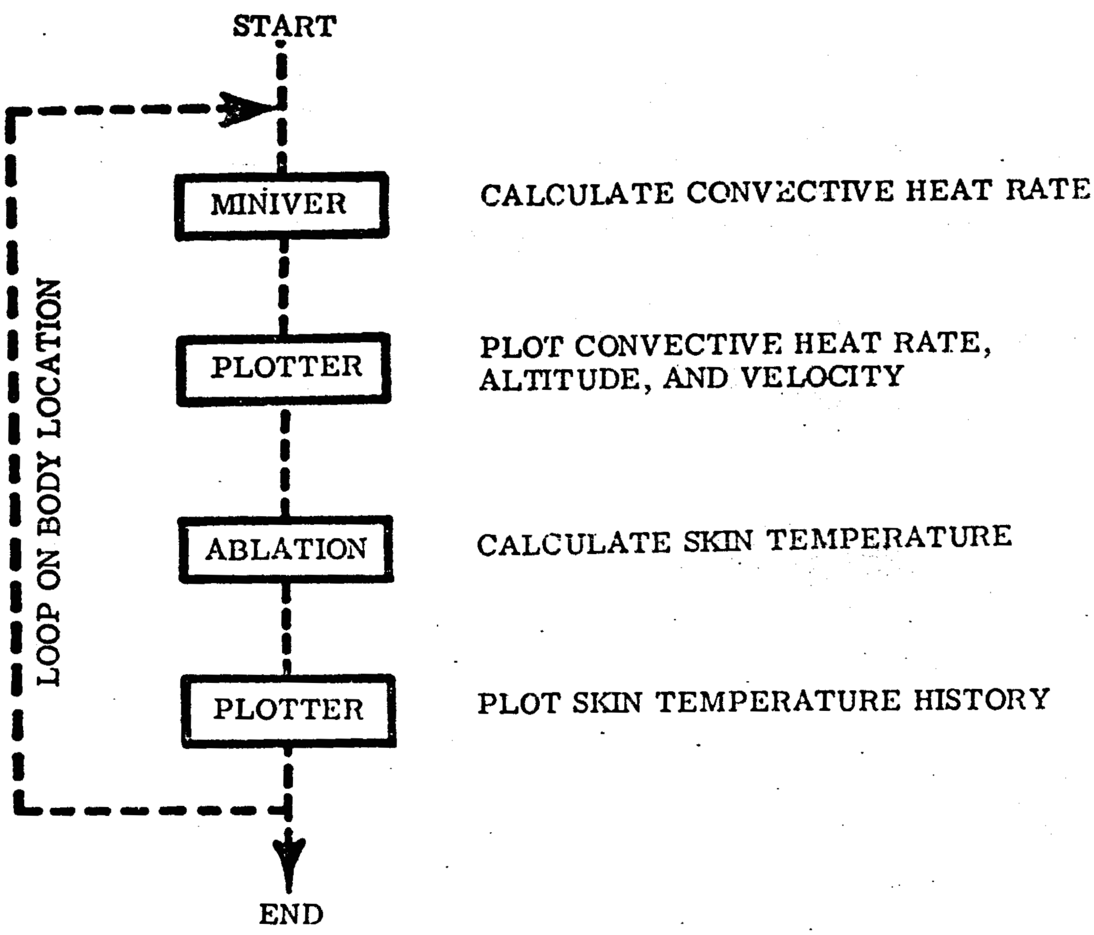
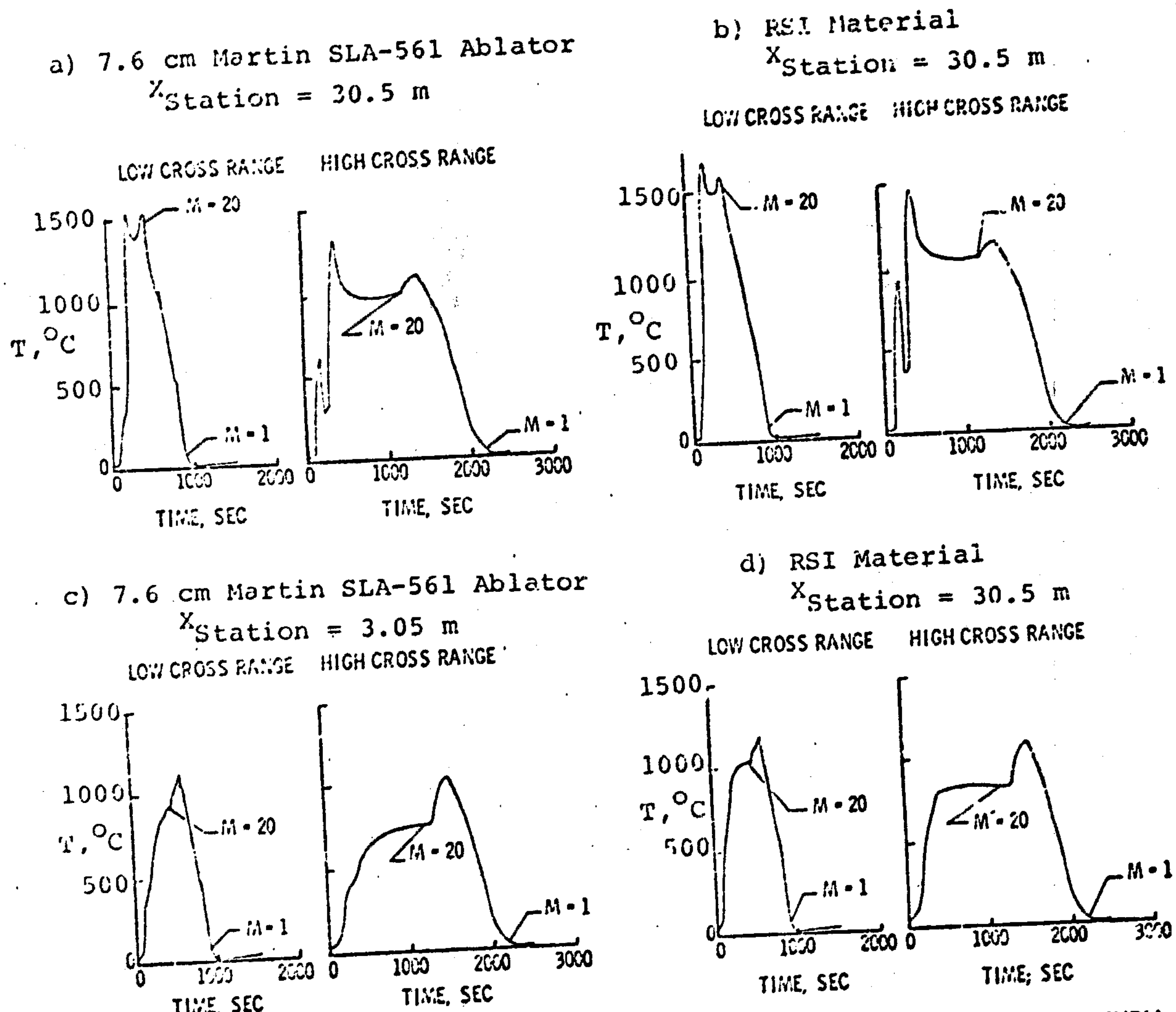


FIGURE 19 ORBITER SURFACE TEMPERATURE STUDY.

over the orbiter surface. Formulation of the problem required the combined efforts of a group of engineers with technological backgrounds in materials, flight dynamics and aerothermodynamics. The problem was solved approximately one week after its conception. The results, shown in figure 20, indicated that no excessive temperatures were present on the orbiter skin during approach and landing using either reusable or ablative material.



RESULTS OF THE ORBITER SURFACE TEMPERATURE STUDY.

CONCLUDING REMARKS

A very large scale synthesizing system for engineering processes has been described. The elements of the system are a library of independent applications computer programs, an executive computer program and a data base which forms the common information link among the independent applications program. The system can be used by individuals for small problems or the operation can employ the design team approach for larger and more complex problems. In the latter case, the design team defines the program sequences, data interfaces and matching loops required to achieve the desired design objective. The simulation is formulated in the language of the executive control program. The system provides the users with the ability to formulate the computer aided design problem at the task level in much the same manner as is employed in the industrial design process.

The executive computer program ODINEX controls the sequence of execution of the independent program elements based on user supplied commands and performs the data management function through the maintenance of the data base of common information which is accessed at the input/output level of the library programs. Each program is executed sequentially and as such is "unaware" of its contribution to a larger and more comprehensive engineering process. ODINEX interrogates the data into and out of the independent programs and performs data manipulations according to "instructions" embedded in the data. The instructions are user supplied and form the control and communication language which are input to the ODINEX executive program. ODINEX restructures the input stream based on the instructions. The result is a flow of information which is identical to the normal flow of input to the individual program executions.

The greatest single advantage of the ODIN system is that it allows full use of virtually all past developments in engineering technology for the synthesis of engineering processes with little or no modification. Any existing checked out computer code can be easily incorporated into the system library and the developer of the new technological modules is unconstrained by requirements of the ODINEX executive system. Little programming knowledge is required to incorporate a program into the system for the first time.

The control and communication language consists of a simple and easily understood set of instructions which provide the capability of creating a network of computer programs for analysis at any level of detail. All synthesis processes and data intercommunication are performed at the program input level. Conditional branching logic is provided for creating sizing and/or optimization loops within the synthesis. There is no effective limit to either the number of programs used or the complexity of design loops created.

The manual data transfer from technology to technology may be drastically reduced using the ODIN system. Further, the chance of data error, data misunderstanding or data misrepresentation is virtually eliminated. All factors dealing with "engineering judgment," "design margins" and "non-optimum analysis" may be employed and are visible to the design team. Data visibility has been a key requirement in the development of the ODIN system. Therefore, report generation is an integral part of ODINEX. User generated reports based on data base information can be generated at any point in the sequence of program executions. A variety of graphical capability is available in the program library to supplement the written reports.

Finally, the ODIN system provides a true building block approach to the synthesis of enginering processes. Technology programs may be added, deleted or replaced to suit the design objective. This provides a responsiveness of computer aided design techniques never before available to the designer. All or any part of the design process may be synthesized when using the ODIN system. A number of examples of the solution of real problems have demonstrated that ODIN can support partial or complete design studies at several levels of design activity.

REFERENCES

1. Gregory, T. J., Peterson, R. J. and Wyss, J. A.: Performance Trade-Offs and Research Problems for Hypersonic Transports. AIAA Journal of Aircraft. July-August 1965.

2. Peterson, R. H., Gregory, T. J. and Smith, C. L.: Some Comparisons of Turboramjet-Powered Hypersonic Aircraft for Cruise and Boost Missions. Journal of Aircraft. September-October 1966.

3. Gold, R. and Ross, S.: Automated Mission Analysis Using A Parametric Sensitivity Executive Program. AAS Paper 68-146, presented at the AAS/AIAA Astronautics Specialist Conference. September 1968.

4. Wennegal, G. J., Mason, P. W. and Rosenbaum, J. D.: IDEAS: Integrated Design and Analysis System. SAE Paper 68-0728, presented to SAE Aeronautics and Space Engineering Meeting. October 1968.

5. Lee, V. A., Ball, H. G., Wadsworth, E. A., Moran, W. J. and McLead, J. D.: Computerized Aircraft Synthesis. AIAA Journal of Aircraft. September-October 1967.

6. Herbst, W. B. and Ross, H.: Application of Computer Aided Design Programs for the Management of Fighter Development Projects. AIAA Paper 70-364, presented at the AIAA Fighter Aircraft Conference. March 1970.

7. Hague, D. S. and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles - ODIN/MFV. AFFDL-TR-72-132. 1973.

8. Glatt, C. R., Hague, D. S. and Watson, D. A.: ODINEX: An Executive Computer Program for Linking Independent Computer Programs. NASA CR-2296. 1973.

9. Rau, Timothy R. and Decker, John P.: ODIN: Optimal Design Integration System for Synthesis of Aerospace Vehicles. AIAA Paper No. 74-72. AIAA 12th. Aerospace Sciences Meeting. 1974.

10. Love, Eugene S.: Advanced Technology and the Space Shuttle. Astronautics and Aeronautics, Volume II, No. 2. February 1973.

11. Henderson, Arthur, Jr.: Aerothermodynamic Technology for Space Shuttle and Beyond. AIAA Paper No. 73-59, presented at the AIAA 9th. Annual Meeting and Technical Display. January 1973.

12. Phillips, W. Pelham, Decker, John P., Rau, Timothy R. and Glatt, C. R.: Computer Aided Space Shuttle Orbiter Wing Design Study. NASA TN D-7478. 1973.

13. Harris, R. V., Jr.: An Analysis and Correlation of Aircraft Wave Drag. NASA TM X-947. 1974.

14. Norton, P. and Glatt, C. R.: VAMP: A Computer Program for Calculating the Volume, Area and Mass Properties of Aerospace Vehicles. NASA CR-2419. 1974.

15. Vanco, Michael: Computer Program for Design Point Performance of Turbojet and Turbofan Engine Cycles. NASA TM X-1340. 1967.

16. Fishbach, Laurence H. and Koenig, Robert W.: GENENG II - A Program for Calculating Design and Off-Design Performance of Two and Three Spool Turbofans with as Many as Three Nozzles. NASA TN D-6553. 1972.

17. Wilwerth, R. E., Rosenbaum, R. C. and Chuck, Wong: PRESTO: Program for Rapid Earth to Space Trajectory Optimization. NASA CR-158. 1965.

18. Stein, L. H., Matthews, M. L. and Frenk, J. W.: STOP - A Computer Program for Supersonic Transport Trajectory Optimization. NASA CR-793. 1967.

19. Kinsey, Don W. and Bowers, Douglas L.: A Computerized Procedure to Obtain the Coordinates and Section Characteristics of NACA Designated Airfoils. Technical Report AFFDL-TR-71-87. Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio. 1971.

20. Hague, D. S. and Glatt, C. R.: An Introduction to Multi-Variable Search Techniques for Parameter Optimization. NASA CR-73200. 1968.

21. Swann, R. T., et. al.: One-Dimensional Numerical Analysis of the Transient Response of Thermal Protection Systems. NASA TN D-2976. 1965.

22. Boren, H. E.: DAPCA: A Computer Program for Determining Aircraft Development and Production Costs. Rand Corporation Report RM-5221-PR. 1967.

23. Glatt, C. R.: IMAGE: A Computer Code for Generating Picture-Like Images of Aerospace Vehicles. NASA CR-2430. 1974.

24. Glatt, C. R., Hague, D. S. and Reiners, S. J.: Prediction of Sonic Boom from Experimental Near-Field Overpressure Data, Method and Results. NASA CR-2441. 1974.

25. Glatt, C. R., Reiners, S. J. and Hague, D. S.: Prediction of Sonic Boom from Experimental Near-Field Overpressure Data, Data Base Construction. NASA CR-2442. 1974.

26. Glatt, C. R.: WAATS: A Computer Program for Weights Analysis of Advanced Transportation Systems. NASA CR-2420. 1974.

27 Glass, K. J. and Whitnah, A.M.: Static Aerodynamic Characteristics of the MSC-040A Space Shuttle Orbiter with Wedge Centerline Vertical and Twin Vertical Tails at Mach Numbers from 0.6 to 4.96. DMS-DR-1243 (NAS 8-4016), Space Division, Chrysler Corporation. March 1972. (Available as NASA CR-120050.)

APPENDIX A: BASIC ODIN PROGRAM ELEMENTS

The independent program elements which form the ODIN library are described in abstract form in this appendix. They include technology programs and utility programs. Each program in the system has been assigned a mnemonic descriptor which is used to activate the program during an ODIN simulation. The organization of the abstracts is alphabetically arranged by the mnemonic descriptor. Further documentation is generally available as NASA publications or aerospace company reports.

The following table gives an index to the program abstracts by technology or function. Many of the programs overlap technologies so they appear more than once in the table. Each technology area (i.e. Aerodynamics, Weights, etc.) will have several programs identified by mnemonic description and program title. The corresponding abstract may be reviewed by looking up the appropriate mnemonic alphabetically in the pages which follow the table.

TABLE OF PROGRAM TITLES BY TECHNOLOGY

EXECUTIVE

- ODINEX: A Computer Program for Linking Other Computer Programs.

GEOMETRY

- AIRFOIL: A Program for Generating Geometric and Aerodynamic Characteristics of Airfoil Sections.
- GEOMETRY: Body Coordinate Generator.
- PANEL: A Program for Generating Panelled Configuration Geometry.
- VAMP: Volume, Area and Mass Properties.
- WETED: Wetted Area in Reference Length Program.

AERODYNAMICS

- AIRFOIL: A Program for Generating Geometric and Aerodynamic Characteristics of Airfoil Sections.
- DATCOM: Configuration Design Analysis Program (TRW).
- DATCOM2: Configuration Design Analysis Program (MDAC).
- HABACP: Hypersonic Arbitrary Body Aerodynamic Computer Program.

SBOOM:	Sonic Boom Prediction for Shuttle Type Vehicles.
SKINF:	Turbulent Skin Friction Drag Program.
TREND:	Subsonic/Supersonic/Hypersonic Aerodynamic Trade-Off Program.
WDRAG:	Zero-Lift Wave Drag Program.

PROPULSION

ENCYCL:	Design Point Performance of Turbojet and Turbofan Engine.
GENENG:	A Program for Calculating Design and Off-Design Performance for Turbojet and Turbofan Engines.
GENENGII:	A Program for Calculating Design and Off-Design Performance of Two and Three Spool Turbofans with as many as Three Nozzles.

WEIGHTS

PADS:	Performance Analysis and Design Synthesis Program.
SSSP:	Space Shuttle Synthesis Program.
VAMP:	Volume, Area and Mass Properties.
VSAC:	Vehicle Synthesis for High Speed Aircraft.
WAATS:	Weights Analysis for Advanced Transportation Systems.

PERFORMANCE

ATOP:	Atmospheric Trajectory Optimization.
COAP:	Combat Optimization and Analysis Program.
PADS:	Performance Analysis and Design Synthesis Program.
POST:	A Program to Optimize Simulated Trajectories.
PRESTO:	Program for Rapid Earth-to-Space Trajectory Optimization.
SEPARATE:	Program to Simulate Separation of Two Stages of a Launch Vehicle.
SSSP:	Space Shuttle Synthesis Program.
TOLAND:	Take-Off and Landing Program.
TOP:	Trajectory Optimization Program.
VSAC:	Vehicle Synthesis for High Speed Aircraft.

STABILITY AND CONTROL

ACMOTAN:	Linear Aircraft Motion Analysis.
DATCOM2:	Configuration Design Analysis Program.
HABACP:	Hypersonic Arbitrary Body Aerodynamic Computer Program.
TREND:	Subsonic/Supersonic/Hypersonic Aerodynamic Trade-Off Program.

THERMODYNAMICS

ABLATOR:	One Dimensional Analysis of the Transient Response of Thermal Protection Systems.
ATOP:	Atmospheric Trajectory Optimization.
HABACP:	Hypersonic Arbitrary Body Aerodynamic Computer Program.
MINIVER:	Aerodynamic Heating Program.
TREND:	Subsonic/Supersonic/Hypersonic Aerodynamic Trade-Off Program.

STRUCTURES

AFSP:	Automated Flutter Solution Procedure.
SSAM:	Swept Strip Aeroelastic Model.

ECONOMICS

DAPCA:	Development and Production Cost of Aircraft.
PRICE:	A Program for Improved Cost Estimation.

OPTIMIZATION

AESOP:	Automated Engineering and Scientific Optimization Program.

GRAPHICS

CONPLOT:	Aircraft Configuration Plot.
HABACP:	Hypersonic Arbitrary Body Aerodynamic Computer Center.
IMAGE:	Configuration Display Program.
IMG250:	Picture Generator for CDC 250 CRT.
PLOTTER:	An Independent Plotter Program.
PLT250:	Independent Plot Program for CDC 250.

PLTVIEW:	Program for Generating Separation Plots.
POSTDATA:	POST Plot Data Generation Program.
TOPLOT:	Plot Generator for TOP

BALANCE

VAMP:	Volume, Area and Mass Properties.

UTILITIES

ABNORM:	Abnormal End Program.
AUTOLAY:	A Program for Automatically Constructing Overlayed Computer Programs.
CCSAVE:	Control Card Data Base Save Procedure.
CHGABEN:	Abnormal End Procedure.
CHGDLOG:	DIALOG Execution Procedure.
COGO:	Run Time Compiler Program.
COLOGO:	Compile, Load and Go Procedure.
COMPILER:	Program to Compile a FORTRAN Program.
DROPEXS:	System File Reduction Procedure.
EDIT:	CRT Editing Program.
ENDODN:	End ODIN Procedure.
FNT:	File Name Table Program.
HEADING:	Heading Program.
JOBTIME:	Simulation Timing Program.
MAC:	Macro-Processor.
MYPROGRAM:	Execute a Compiled Program.
PLOTSV:	Procedure for Saving CALCOMP Plots.
PRINTER:	Procedure for Printing Previously Generated Output Information.
REPORT:	Internal Procedure for Generating an Engineering Report.
ROUTECC:	Data Routing Procedure.
ROUTEXP:	Dynamic Printing Procedure.
RSTART:	Restart Procedure.
VARIAN:	VARIAN Plot Processing Procedure.

ABLATOR: One Dimensional Analysis of the Transient Response of Thermal Protection Systems.

The program assumes that thermal properties in the given layer of material are functions only of temperature, that all heat flow is normal to the surface, and that gases transpiring to the char layer are the same temperature as the char. The outer surface is subjected to the aerodynamic heating and the char layer provides both installation and high temperature outer surface for reradiation. The heat passing through the layer is partially observed by pirolysis at the interface between the char layer and the uncharedmaterial. The remaining heat is conducted into the uncharred layer. The gases generated through the char layer are injected into the boundary layer. The gases are heated as they pass through the char and this heat removal from the char layer induces the quantity of heat conducted by the pirolysis interface. When these gases are injected into the boundary layer, the conducted heat transfers are induced. The program accepts as input the temperature at the surface of the model and generates a time history of the thermal condition of the various layers as the vehicle enters the atmosphere.

ABNORM: Abnormal End Program.

The program is used as part of the abnormal end procedure in ODIN. It interrogates system files to determine whether a plot vector data file has been written, the program generates plot request cards and disk-to-tape copying instructions for the files with appropriate information on them. The tapes are assigned to the user, whose name is on the JOB card.

ACMOTAN: Linear Aircraft Motion Analysis

The program is a versatile code for linear aircraft motion analysis which allows the user to supplement the standard airplane equations of motion with auxiliary equations written by the user to represent control laws or additional variables. The program prepares the system of linear differential equations using several optional forms of input data and then carries the solution to an extent determined by the output option selected. Minimum output includes the characteristic polynominal and its roots. Additional output in the form of transfer functions, frequently responses and time histories can be selected.

AESOP: Automated Engineering and Scientific Optimization Program.

AESOP is a multiple variable optimization program designed for the solution of a wide range of parameter optimization problems The basic program has the ability to solve constrained optimization problems involving up to 100 parameters and up to 20 constraints. Thirteen search techniques are available for use individually or in combination to solve the desired problem. The methods include sectioning, steepest descent, quadrilateral search, Davidon's method, random ray search, pattern and severa others. The program is designed to be linked with other progra to perform internal optimization or can be used as an independe program for optimizing systems of programs.

AFSP: Automated Flutter Solution Procedure

Program AFSP is based upon a new method of solving the flutter equation. The method is based upon the premise that the flutte analysis, due to the particular form in which the aerodynamic forces are available, essentially consists of a search for thos V-values and w-values which render the flutter determinate zero The procedure deviates essentially from the V-g analysis in so far that the eigenvalue calculation is replaced by simplier algorithm leading to the decomposition of the flutter matrix. The actual search for the flutter solution involves a single re and positive function of the two variables V and w rather than the (n) complex functions representing the eigenvalues in the V analysis. The flight altitude, Mach number and flight speed re main consistent throughout the search whereas the V-g analysis starts out from given values at altitude and Mach number. The flutter speed only follows as a result of the calculation. In general, the speed will not be consistent with the input data such that many runs are required to iterate toward a solution.

AIRFOIL: A Program for Generating Geometric and Aerodynamic Characteristics of Airfoil Sections.

The computer program is written to provide airfoil coordinates incompressible inviscid section characteristics and two-dimensi al drag-rise Mach numbers for a large number of National Advisc Committee for Aeronautics (NACA) designated airfoils from a sir one card input. The program is actually a combination of two separate programs. One program gives the airfoil surface coorc nates with only the NACA airfoil designation as input, and the other uses the surface coordinates to predict incompressible, inviscid pressure distribution from which the section charactei istics and drag-rise Mach number are determined.

ATOP: Atmospheric Trajectory Optimization

The program is a generalized steepest descent computer program set up to handle the three-dimensional, point mass, vehicle flight path trajectory optimization problem. It is capable of simultaneously handling up to fifteen state variables, six control variables and ten constraints. Most of the usual functions required in flight path studies are available within the program; others may be added as desired by simple program additions, providing the function or its derivative is defined analytically. The program may be readily extended to cover steepest descent optimization problems in other fields, by the replacement of the basic differential equation subroutine by any other set of equations of the same general type. Convergence to the optimal solution is obtained automatically by means of one of two control systems which, by a series of logical decisions, obtain a reasonable perturbation magnitude at each iteration.

AUTOLAY: A Program for Automatically Constructing Overlayed Computer Programs.

AUTOLAY is a CDC 6600 utility program used in the construction of overlay files from relocatable binary program elements of the type generated by the FORTRAN compiler.

The use of AUTOLAY permits the construction of a new program file from one to six library files. The new program file contains the user selected main program or programs together with all the subroutines (external references) which the main program (s) call. The main program (s) are specified by the user in a text file read by AUTOLAY.

CCLINK: Control Card Linkage Program.

CCLINK is a CDC 6600 utility program which performs the linking of control card sequences under direction of the ODIN executive program. This software package directs the system to read control cards from alternate files (names on the CCLINK control card). Although developed specifically to meet the linkage requirement of the ODIN system, the CCLINK software can be used as an independent utility program. CCLINK is part of the computer operating system at Langley Research Center and is accessed by a control card or from a FORTRAN program.

CCSAVE: Control Card Data Base Save Procedure.

CCSAVE uses the computer operating system software to save a newly created or updated control card data base. The control card data base is a data base of execution procedures for all the ODIN library programs and procedures.

CHGABEN: Abnormal End Procedure.

The CHGABEN procedure permits the user to alter the control card procedure executed in case of an abnormal end during an ODIN simulation. The preprogrammed procedure copies the input file, the output file and the control file from the previously executed or executing program to the output file and saves any previously generated plot information on physical tape. The procedure may be changed by providing the desired control card procedure as data input.

CHGDLOG: ODINEX Execution Procedure.

CHGDLOG permits the user to replace the ODINEX execution procedure. The preprogrammed ODINEX executive procedure is usually satisfactory for most simulations. However, under certain circumstances, it may be desirable to alter the procedure for one or more technology program executions during a simulation sequence. The desired procedure is provided as data input.

COAP: Combat Optimization and Analysis Program.

The program is an extension of the ATOP II (Atmospheric Trajectory Optimization Program). It uses two complete three dimensional equations of motion sets to simulate a one-on-one combative encounter between two flight vehicles. The aerodynamic and propulsion representation are sufficiently general to permit the simulation of both current and proposed vehicles by input data. Generalized rotating planetary and atmospheric models permit stimulation of either aircraft, missiles or spacecraft encounters. Combat roles for each vehicle (attacker, defender, etc.) are automatically defined on the basis of vehicle relative positions, heading and velocities. Depending upon the vehicle role selected, any one of the set of tactics designed to satisfy the role requirement is executed. The tactics vary in nature from straight forward stylized maneuvers such as split-S to barrel role, to three-dimensional lag or lead pursuit path. Combat optimization capability may be introduced by repetitive simulation using parameterization of the combat guidance parameters and the application of multivariable search techniques. Alternately, the variational calculus may be employed to define optimal continuou control against a reacting opponent. In the parameter optimization mode, the option to determine a mini-max solution is also available.

COGO: Run Time Compiler Program.

COGO is an independent program that accepts FORTRAN equations as input, compiles the equation and performs the indicated FORTRAN operation. For each input equation, the computed result is plac in the ODIN information file for potential storage in the dynami data base. The COGO program uses a run time compiler developed for Langley Research Center. The principle of operation of the run time compiler is similar to that of the system compiler except each line or equation is executed as it is compiled. Only the real variables may be employed and no branching logic is provided.

COLOGO: Compile, Load and Go Procedure.

The COLOGO procedure is a combination of the COMPILER and MY-PROGRAM procedures. The compile and execute functions are performed by a single directive using the system run compiler and the system loader. The procedure does not permit the use of dat reads directly. However, the program can obtain data from the data base by the placement of data base interface information directly into the FORTRAN program itself.

COMPILER: Program to Compile a FORTRAN Program.

This procedure uses the run compiler to compile a user supplied FORTRAN program within the run stream to an ODIN simulation. Th input data is a standard FORTRAN program which can be augmented by data base information such as for variable dimensions, data statements, etc. To execute the compiled code the user executes the MYPROGRAM procedure. The COMPILER procedure gives the ODIN user the capability of performing complex data transformations using the full capability of the FORTRAN language.

CONPLOT: Aircraft Configuration Plot.

This program generates the necessary instructions for automatically plotting of a numerical model of an aircraft configuratio: Program options may be used to draw three-view and oblique orth graphic projections as well as perspective drawings of an aircr These plots are useful in checking the accuracy of the numerica model data. Magnetic tape output from this program has been us to drive the CALCOMP plotter and a GERBER plotter. The program has also been used for online display of a configuration on the CDC 250 Cathode Ray Display Device.

DAPCA: Development and Production Cost of Aircraft.

The DAPCA program computes development and production costs of major subsystem of fly away aircraft (airframe, engines, etc.). The cost input data is simple and generally relates to aircraft and engine performance characteristics such as gross takeoff weight, speed engine type thrust, etc. The actual cost equatior are the power law types with built-in cost coefficient and usinc user supplied parameters.

DATCOM: Configuration Design Analysis Program (TRW).

The program computes aerodynamic coefficients for aircraft/space craft configurations in the subsonic/transonic/supersonic regime Analytical techniques in the program are based on those of USAF stability and control handbook, DATCOM, revision September, 1970 The program comprises four modules which compute lift, pitch, sid slip and control characteristics respectively. Modular construc tion enables other sets of aerodynamic characteristics to be incorporated into the program.

DATCOM2: Configuration Design Analysis Program (MDAC).

The program calculates the statics stability characteristics of wings, bodies, wing-body, tail-body and wing-body-tail combinations at angle of attack and slide slip through the Mach range from subsonic to supersonic speeds. Whenever appropriate DATCOM methods are available, the program computes longitudinal derivatives C_{L_α} and C_{m_α}, longitudinal coefficient C_D, C_L, C_m and side slip derivates C_{Y_β}, C_{l_β} and C_{n_β}. Output for configurations of horizontal tails also include downwash and dynamic pressure valu All intermediate variable calculations are also available for output.

DROPEXS: System File Reduction Procedure.

DROPEXS is a procedure for purging system files which are used during the initialization of the ODINEX executive system but not used during execution of the simulation. DROPEXS was written to reduce system file requirements and make available space in the system file name table for other uses.

EDIT: CRT Editing Program.

The program was developed for users of the CDC 250 CRT scope system at LRC to accommodate online source file editing recompilation and re-execution. With the proper use of control cards, it also has the capability to recover from abnormal exits. Whereupon certain types of corrections can be made and the program can be restarted. The EDIT provides four modes of display, source, input, output and record. Source is the users source language program. Input is the users input control card file. Output is the users normal output file and record is any record of a system file which is attached to the current job. The EDIT program is used in the ODIN system for data verification between technology module execution when operating ODIN in the online mode.

ENCYCL: Design Point Performance of Turbojet and Turbofan Engine.

ENCYCL computes the design point performance of turbojet and turbofan engine cycles from user supplied engine characteristics and flight conditions. The program input requires the airplane Mach number, the altitude, the state conditions, turbine inlet temperature, afterburner temperature, duct burner temperature, bypass ratio, coolant flow, component efficiencies and component pressure ratios. The output yields specific thrust and specific fuel consumption, engine efficiencies and several component temperatures and pressures. The thermodynamic properties of the gas are expressed as functions of the temperature and fuel to air ratio.

ENDODN: End ODIN Procedure.

ENDODN is a user supplied procedure which is executed at the normal termination of the ODIN simulation. Usually the procedure is used for saving the dynamic portion of the ODIN data base for future use. However, any supplemental control procedure can be supplied by the user. ENDODN does not require a special execution command.

FNT: File Name Table Program.

The program generates a tabular list of files currently attached to the job currently in execution. It also determines the number of files attached to control point zero (assigned to the system) and the number of names available in the file name table. It is used in ODIN to isolate unused files created during execution.

GENENG: A Program for Calculating Design and Off-Design Performance for Turbojet and Turbofan Engines.

The program calculates steady state design and off-design performance for one and two spool turbojet engines. The original version of the GENENG program entitled, "Simulation of Turbofan Engines" was developed by the Turboengine Division of the Air Force Aero Propulsion Laboratory, Wright Patterson Air Force Base, Ohio. The program uses steady state gas dynamics to compute the engine design conditions. Off-design performance is based on specific component performance maps which must be provided by the user.

GENENGII: A Program for Calculating Design and Off-Design Performance of Two and Three Spool Turbofans with as many as Three Nozzles.

The GENENGII program is a derivative of GENENG (Generalized Engine Program). GENENG is capable of calculating steady state design and off-design performance of turbofan and turbojet engines were evolved from SMOTE (Simulation of Turbofan Engine) which was developed by the Turbo Engine Division of the Air Force Aero Propulsion Laboratory of Wright Patterson Air Force Base, Ohio. GENENGII calculates design and off-design engine performance for existing or theoretical fan engines with two or three spools and with one, two or three nozzles. In addition, fan performance can also be calculated. Nine basic turbofan engines can be calculated without any programming changes. Included among the nine are three types which are likely candidates for STOL aircraft with internally blown flaps. Many other possibilities exist which are too numerous to mention, being determined by the users knowledge of the program itself.

GEOMETRY: Body Coordinate Generator.

The program generates trapezoidal and elliptical body coordinates in a format suitable for use in VAMP, WETTED, DRAG and CONPLOT. The forbody coordinates are generated according to a "minimum drag" area distribution while the afterbody coordinates are constant in cross sectional area.

GOGET: Modified Data Cell Retrieval Program.

GOGET is a modified version of the LRC computer system FETCH program for retrieving information from data cells. GOGET allows the user to retrieve data sets from data cell without the use of the wedge number. The usage is identical to the FETCH program except that the wedge number position parameter on the control card is omitted.

HABACP: Hypersonic Arbitrary Body Aerodynamic Computer Program.

The program treats the vehicle surface as a collection of quadrilateral elements oriented tangential to the local vehicle surface. Each individual panel may have its local pressure coefficient specified by any of a variety of pressure calculation methods, including modified Newtonian, blunt body, Newtonian-Prandtl-Meyer, tangent-wedge, tangent-cone, boundary layer induced pressures, free molecular flow and a number of empirical relationships. Viscous forces may also be calculated, which include viscous-inviscid interaction effects. Skin friction options include the reference temperature and referenced enthalpy methods for both laminar and turbulent flow, the Spalding-Chi method and a special blunt body skin friction method. Control surface deflection pressures including separation effects that may be caused by the deflected surface are also calculated. Several other options are available including the calculation of dynamic derivatives, the generation of geometry and plotting.

HEADING: Heading Program.

HEADING is a control card callable program which prints user specified heading information in large characters. The characters of the heading are formed by the pattern of characters which form the character itself. The letters of the heading are eight characters wide and ten characters high. Input to the program is placed on the heading execution card.

IMAGE: Configuration Display Program.

The IMAGE program uses a surface definition based on quadrilateral elements to describe picture-like drawings of arbitrary configurations. The program is used for visual check on geometric input data, monitoring of geometric perturbations and providing reports on geometric characteristics. Geometric characteristics may be input or taken from a data base of configuration data. The user describes the viewing angles, position and scaling factors as well as textual information through the input procedure. The configuration drawings are generated on a plot vector file which is suitable for processing by the CALCOMP processor.

IMG250: Picture Generator for CDC 250 CRT.

The program is identical in structure to the IMAGE program for generating picture-like images on offline devices. IMG250 generates pictures (from the same data) for the CDC 250 CRT. IMG250 is used for editing geometric data when operating the ODIN system in the online mode.

JOBTIME: Simulation Timing Program.

JOBTIME generates a day file message which specifies the number of CPU seconds and the number of operating system requests (OS CALLS) accumulated since the start of the job.

MAC: Macro-Processor

MAC is a precompilation type string processing language generator which permits the user to construct new languages or augment existing ones. Input command structures can be developed which are later broken down by the MAC program and converted into language elements for processing by existing compilers or data processors.

MINIVER: Aerodynamic Heating Program.

The program is a "miniature version" of the McDonnell Douglas Corporation Aerodynamic Heating Program for use on the CDC 6600. It calculates radiation equilibrium temperature and provides thin-skin temperature response for input materials. (Aluminium, titanium, Rene 41 and inconnel X-750 properties are builtin.) Heat transfer methods are available including Fay and Rydell, Erkart Reference Enthalpy, Spaulding and Chi, Flat Plate Rho-Mu product, three swept cylinder theories and Lees-Detra and Hidalgo for hemispheric nose caps. The program also includes the capability of traversing three sequential shocks, shape-edge and cone plus oblique shock of availability, sharp-edge and cone modified Newtonian and swept cylinder stagnation line pressure solution.

MYPROGRAM: Execute a Compiled Program.

This procedure is used to execute a compiled FORTRAN program, which has been generated by the COMPILER procedure. It is separated from the compiler procedure so that the user may do a one-time compilation early in a simulation and use the compiled program many times later in the simulation. Input data is provided by the user in the format specified in the COMPILER generated program.

ODINEX: A Computer Program for Linking Other Computer Programs.

ODINEX is an executive computer program which controls the sequence of execution and data management function for a library of independent computer programs. Communication of common information is accomplished by ODINEX through a dynamically constructed and maintained data base of common information. The unique feature of the ODINEX executive system is the manner in which computer programs are linked. Each program maintains its individual identity and as such is unaware of its contribution to the large scale program. This feature makes any computer program a candidate for use with the ODINEX executive system.

PADS: Performance Analysis and Design Synthesis Program.

The Performance Analysis and Design Synthesis (PADS) computer program has a two-fold purpose. It can size launch vehicles in conjunction with calculus of variations optimal trajectories and can also be used as a general purpose branched trajectory optimization program. In the former case, it has the Space Shuttle Synthesis Program as well as a simplified stage weight module for optimally sizing manned recoverable launch vehicles. For trajectory optimization alone or with sizing, PADS has two trajectory modules. The first trajectory module uses the method of steepest descent, the second employs the method of quasilinearization, which requires a starting solution from the first trajectory module.

PANEL: A Program for Generating Panelled Configuration Geometry.

The PANEL program is a general purpose external geometry definition program developed primarily for use in large scale simulations of the preliminary design process. It is an independently operated program which produces a sequence of quadrilateral panels defined by the corner points. The resulting data is acceptable as input to computer programs in other technical disciplines such as aerodynamics, structures and thermodynamics.

The program accepts as input a variety of formats from detailed definition of individual panels to selection of generalized two-and-three dimensional shapes. The section data includes circular and elliptical as well as arbitrary cross sections. A cubic patch technique is included which allows broad sections of the vehicle to be described with a relatively small amount of input. The input data can be mathematically fitted with end matched cubic functions and reduced to distributed panels.

PLOTSV: Procedure for Saving CALCOMP Plots.

The procedure is used for collecting previously written CALCOMP plot files on a physical tape for submittal to the CALCOMP plot hardware. The ODIN system is designed to execute many programs, any of which may produce plotted information. Provisions have been made by which plot vector data is accumulated on a temporary disk file during a given ODIN run. This provision avoids the mounting of physical tapes during a simulation. Upon execution of PLOTSV, the plotted information is transferred to a physical tape and the tape is unloaded for later plotting.

PLOTTER: An Independent Plotter Program.

The program provides a generalized X-y plotting and contour drawing capability for use with the program. Plot data may come from the input stream or from auxiliary files created by other program elements. The plot program can generate plot vector files for a variety of plot devices including CALCOMP Gerber, CRT and the printer.

PLT250: Independent Plot Program for CDC 250.

The program is an independent plot program which accepts data to be plotted from the input or from plot f.le generated by another program. The input PLT250 is identical with the input for PLOTTEI The output is generated on the CDC 250 scope rather than an off-line device.

PLTVIEW: Program for Generating Separation Plots.

The program is a companion to the SEPARTE program which computes the separation distance and attitude of two stages of the shuttle vehicle during staging. PLTVIEW generates plots of separation distance and vehicle orientation for the separating vehicle components. The input data is obtained from the time history data generated by the SEPARTE program. The output is a GERBER plot of the separation distance and orientation.

POST: A Program to Optimize Simulated Trajectories.

The POST program is a generalized point mass discrete parameter targeting an optimization program. POST provides the capability to target and optimize point mass trajectories for a powered or unpowered vehicle near an arbitrary rotating oblate planet. POST has been successfully used in solving a wide variety of atmospheric ascent and re-entry problems as well as exoatmospheric orbital transfer problems. The generality of the program is evidenced by its N-phase simulation capability which features generalized planet and vehicle models. This flexible simulation capability is augmented by an efficient, discrete parameter optimization capability which includes equality and inequality constraints.

POSTDATA: POST Plot Data Generation Program.

The program interrogates the output data tape from the POST program and generates specified plot information pertaining to time histories of various performance and constraint functions available after the execution of the POST program. The program is designed specifically as an interface program for analysis of trajectory data from the POST program.

PRESTO: Program for Rapid Earth-to-Space Trajectory Optimization.

The PRESTO program uses a closed loop deepest descent optimization procedure to derive flight trajectories to produce maximum booster payloads for a variety of space missions. Trajectories can be computed in three degrees of freedom about a spherical rotating earth. Four powered stages and three upper stage thrust cycles can be accommodated. Coast periods are permitted between each stage. Aerodynamic lift and drag forces are included in the computation. The optimization routine simultaneously considered the launch direction and time the interstage coast durations and the upper stage thrust sequencing, the complete pitch and yaw attitude histories and terminal constraints. Immediate constraints may be introduced on angle attacks, coast orbit perigee altitude or on the product on angle attack and dynamic pressure. The closed loop procedure greatly facilitates the satisfaction of terminal constraints and reduces the number of iterations required to achieve convergence.

PRICE: A Program for Improved Cost Estimation.

The PRICE computer program was developed in order to rapidly generate preliminary estimates of total program costs for mission studies of V-STOL and conventional transport aircraft, hypersonic aircraft and reusable space transportation system. The program uses cost estimating relationships based on historical cost data for conventional and advanced aircraft, spacecraft and launch vehicles. The approach is based on correlating cost with parameters such as weight and thrust, then adjusting the results with complexity factors to account for differences in material and type of construction, performance level, etc.

PRINTER: Procedure for Printing Previously Generated Output Information.

The output from an ODIN program is normally not printed. However the user has the option to obtain the printout through use of the printer procedure immediately following the execution of the ODIN program. The printer program is used to transfer the normal output from a special file generated at execution time (for the ODIN program) to the normal output file.

REPORT: Report Generator.

REPORT is a submodule of the ODINEX program which allows the user to interrogate the data base and format engineering reports. Descriptive information may be provided in any format with delimited data base names inserted where data base requests are desired. The report is automatically printed with descriptive information printed exactly as originally specified and delimited data base names replaced by the corresponding data base information.

ROUTECC: Data Routing Procedure.

ROUTECC is a procedure whereby the user may route he normal output from any ODIN program to the central computer facility for printing on high speed printers. It avoids use of the medium speed printers associated with the batch terminal system at LRC. ROUTECC is executed after each ODIN program for which the printout is desired. The printed output is returned to the originating terminal by the usual delivery services.

ROUTEXP: Dynamic Printing Procedure.

This procedure allows the user to obtain data base information at the originating terminal for ODIN simulations in progress. The input to this procedure is a user formatted report with the desired data base interface requests appropriately paced. The output is a printed report with the interface request replaced with current data base values at the time the report was generate The report is printed immediately after it is generated although the simulation may still be in progress.

RSTART: Restart Procedure.

RSTART is a procedure which affects a reinitialization of the ODINEX executive system and restart of the current ODIN simulation. The input data set name is assumed to be the same as the data with which the simulation was originally started. RSTART is used in the online CRT mode of operation for restart if necessary after execution of the EDIT program and ensuring data alterations.

RTEXT: Data Cell Label Retrieval Program.

The program constructs a data cell label from information contained in the data cell file retrieved by the FETCH or GOGET program. The label is constructed and placed on an auxiliary file called LABEL.

SBOOM: Sonic Boom Prediction for Shuttle Type Vehicles.

The SBOOM program is based on the prediction technique of Thomas (references). The prediction technique calculates the far-field overpressure from the near-field pressure signatures measured in the wind tunnel. The wind tunnel results generally are stored in a data base and accessed by the computer program for interpolation/extrapolation based on geometric similarity of the pressure signatures. Wind tunnel data in the data base was generated for space shuttle type configurations. The approach used in determining ground overpressure is to describe the wave form of the sonic boom wave by several wave form parameters and then obtaining equations for the parameters as function time. This approach has the advantage of being simple and providing a convenient method for extrapolating experimental signatures because the signatures are dealt with directly. The input consists of the vehicle conditions and the environment in which the vehicle is operating as a basic condition operating the program. Many flight conditions can be equated through the normal input channels or the program will accept flight condition data from trajectory generation programs stored on an auxiliary file.

SEPARTE: The Program to Simulate Separating Stages of Launch Vehicles.

The program is a twelve-degree-of-freedom separation program for studying space shuttle separation problems. Each stage of the separating vehicle is represented by six degrees of freedom. Aerodynamic data governing the motion of the separating vehicle is staged into the program from a storage file because of the large bulk of data. The program generates Gerber plots of the separating vehicle components at specified time intervals. Separating components are represented by simplified geometric shapes.

SKINF: Turbulance Skin Friction Drag Program.

The program computes the skin friction drag of a vehicle including the effects of distributed roughness and temperature of the surfaces at arbitrary combinations of Mach number and altitude. Calculations can be made using either the standard day or the +10 degrees hot day atmospheres. Input consists of flight conditions (Mach-altitude), the wetted areas, reference length and form factors for all of the components of the aircraft in the mean roughness height and emittance of the surfaces. Wetted areas in reference lengths may be obtained from the program WETED.

SSAM: Swept Strip Aeroelastic Model.

The program performs an aeroelastic evaluation of the wing spanwise flight load analysis including the complete aircraft balance for a specified set of steady state maneuvers and/or design lift conditions. Here, proper inclusion of the wing-body and the nacelle aerodynamic and weight effects are included in order to compute the required balance tail load which is reflected in the wing load calculation. The flight loads, including the aerodynamic and wing dead weight loads, are converted into structural wing box bending and torsion loads to evaluate the resulting bending and torsional stresses. If the calculated wing stress exceeds the allowable wing stresses, a new set of values of wing section stiffness values are selected to match the allowable stress distribution specified within the program data. The wing aeroelastic load solution is then repeated until the calculated and allowable wing stresses are matched. The cycling process is fast and usually requires three to five cycles to converge depending upon the error margin set within the program. The final calculation is the wing box weight based on the final set of EI and GJ values obtained. The analysis is limited to subsonic flight conditions.

SSSP: Space Shuttle Synthesis Program.

The program automates the trajectory weights and performance computation essential to predesign of the space shuttle system for earth-to-orbit operation. The two-stage space shuttle system is a completely reusable space transportation system, consisting of a booster and an orbiter element. The SSSP major subprograms are detailed weights/volume routine, a precision three dimensional trajectory simulation and the iteration and synthesis logic necessary to satisfy the hardware and trajectory constraints. Three versions of SSSP are available representing some early space shuttle concepts in the predesign stage of the shuttle project.

TOLAND: Take-Off and Landing Program.

The program was constructed by NASA's Advanced Concepts and Mission Division, OART. The program provides simplified high lift aerodynamics based on DATCOM methods, a ground roll analysis, rotation logic and climbout to clear a fifty-foot obstacle. Angle-of-attack in the ground run and rotation maneuvers are determined from the vehicle geometry which is input to the program.

TOP: Trajectory Optimization Program.

A steepest-ascent optimization program has been developed which is capable of optimizing the flight path of a wide class of vehicles. The program will optimize rockets, air-augmented rockets, ramjets, scramjets, turbojets and glide vehicles. Unique features are incorporated which allow extension of optimization procedures into the airbreathing propulsion field, particularly to supersonic transport type vehicles.

The body pitch angle is used for in-plane trajectory shaping in place of the usual angle-of-attack control variable. Additional control variables include bank angle, engine throttling and a variable geometry control variable for vehicles with variable sweep wings, drag brakes, etc. Enroute placards are available to allow optimization within realistic constraints, such as control limits, structural loads, engine operating limits, manned vehicle requirements and geopolitical limitations such as sonic boom.

The optimization is accomplished with an automatic step-size controller and with automatic control variable weighting matrices to allow problem solution in a single computer run. Automatic plotting capability is included. Multistaged vehicles and problems involving variable initial conditions may be optimized. A variable step Runge-Kutta integrator performs the derivative evaluations.

TOPLOT: Plot Generator for TOP.

This program interrogates the output file from TOP (Boeing Trajectory Optimization Program) and generates time history plots of various performance and constraints functions generated during a top run. The output is placed on temporary disk and can be transferred to a physical tape for plotting using the PLOTSV procedure.

TREND: Subsonic/Supersonic/Hypersonic Aerodynamic Trade-Off Program.

The program provides rapid aerodynamic lift, drag and moment estimates and the subsonic, supersonic and hypersonic lift regimes. It is primarily designed to estimate high lift/drag re-entry vehicle aerodynamic characteristic, but the class of vehicles which may be analyzed by the program is of greater range than the primary class of vehicle. Some program modification may be required if the extension is too great. The program may be used for generating basic aerodynamic coefficients or it may be used for trending from known aerodynamic characteristics based on theoretical changes in the geometry. In the hypersonic flight regime, the program contains an optional aerodynamic heating computation capability. The program does not possess a transconic aerodynamic characteristic estimation capability.

VAMP: Volume, Area and Mass Properties.

The VAMP computer program calculates the mass properties, c.g. location, enclosed volume, wetted area and planform area of any closed structure that has a plane of symmetry. The vehicle is described to the computer program by ordered sets of X, Y, Z coordinates of points on its surface. The X, Y, Z coordinates are converted to quadrilateral elements for analysis. The mass properties of each quadrilateral may be computed from a thickness and density input for each quadrilateral or from a weight per unit area input at each point or from a combination of both.

The computed mass property totals may contain not only the contribution from the distributed mass on the vehicle surface wall, but additional masses may be added as "black boxes" by specifying each one's c.g. location and mass properties.

Program VAMP can also produce picture-like images of the vehicle or individual sections from the quadrilateral element data. This facilitates checking the input and visualizing and/or modified configurations. Orthographic, perspective and stero views may be obtained.

VARIAN: Data Transfer Procedure.

VARIAN is a procedure for transferring accumulated plot vector data for the VARIAN plotter from the temporary disk file to a physical tape. The physical tape request card must be provided as data by the user.

VSAC: Vehicle Synthesis for High Speed Aircraft.

The VSAC program was developed for use in preliminary sizing studies of high speed flight vehicles. Single-stage aircraft with either takeoff and landing capability or airborne start, and two-stage vehicles consisting of an aircraft with an expendable, rocket-propelled booster can be sized. The single-stage aircraft are capable of either rocket or airbreathing propulsion and flight speeds to Mach 6. Two-stage configurations cover flight regimes to Mach 15, although no supersonic-combustion engines were considered.

Mission modeling is provided in a modular fashion to allow the program user maximum flexibility, and to allow improvement or substitution of subroutines as new studies may require. Analytic performance equations and numerical trajectory integration are provided, together with appropriate iteration routines.

The primary aerodynamics method is a component buildup procedure that requires the user to input only a geometrical description of the external vehicle characteristics and generates curve-fit expressions for the lift and drag coefficients. Stability and control are not considered.

WAATS: Weights Analysis for Advanced Transportation Systems.

WAATS is a weights analysis program which uses the component buildup technique for weight estimation. Each component weight is based on the weight of the same component of similar vehicles that have actually been built or at least designed in great detail. The similarity law that gives the best correlation for most subsystems has been shown to be the power law formula:

$$y = a\,x^{b}.$$

The program logic assumes the propellant weight and physical characteristics are known. It performs the weight estimations with user supplied correlation parameters (a and b), estimated gross weight and estimated landing weight. An internal iteration loop cycles through the equations until convergence on weight is achieved.

WDRAG: Zero-Lift Wave Drag Program.

The program calculates the zero-lift wave drag at supersonic speeds for airplanes having arbitrary combinations of fuselage, wing, nacelle, horizontal fins and vertical tail. The input geometry description is in the format of CONPLOT and WETTED programs. The program may also be used to calculate (for any desired Mach number) the fuselage area distributions that are required for a minimum wave drag. Simplified fuselage constraints may be included for the above calculation. The fuselage may be cambered and may have arbitrary cross-sectional shapes. The wing may be twisted and cambered. The theoretical approach is based on the far-field linear theory that considered the relationship between the forces on the airplane and the momentum transported through the boundaries of a control surface completely surrounding the airplane.

WETTED: Wetted Area in Reference Length Program.

This program computes the surface wetted area and reference length for all components of a configuration based on corner-point geometry inputs to the program. The configuration may be arbitrarially divided into components on the fuselage and all aerodynamic surfaces. The wing may be divided into many streamwise panels in order to approximate a strip integration for skin friction calculations. Input to the program is identical to CONPLOT. Output is suitable for use in the SKINF program.

APPENDIX B: THE ODIN EXECUTIVE PROGRAM

The ODIN executive program, called ODINEX, is used for linking independent technology computer programs. The linkage forms an interdependent system of automated tools for synthesizing engineering design and analysis processes. The use of the system requires the prior assimilation of the following basic components:

1. A library of independent technology programs including the ODINEX program.

2. The control card sequences for the execution of the independent programs.

3. The setup data for the independent programs which perform the desired analysis function.

The basic components are often available without additional program development. The program sequencing and intercommunication required to integrate the basic components into a design simulation are controlled by ODINEX. The basic capabilities of the program are:

1. Language elements for the construction and maintenance of control card sequences and design data.

2. Language elements for controlling the execution of a sequence of independent programs by simple commands.

3. Language elements for automatically retrieving data base information as input to any of the technology programs.

4. A simple technique for generating summary reports of data base information.

The components of the ODIN system are applied to a design problem as illustrated in figure B-1. Operating on the simulation stream constructed by the user from the input data for the individual technology programs and the executive sequencing/communication instructions, the ODINEX program merges a portion of the simulation stream representing one technology program execution with data base information to form a control card sequence and modified

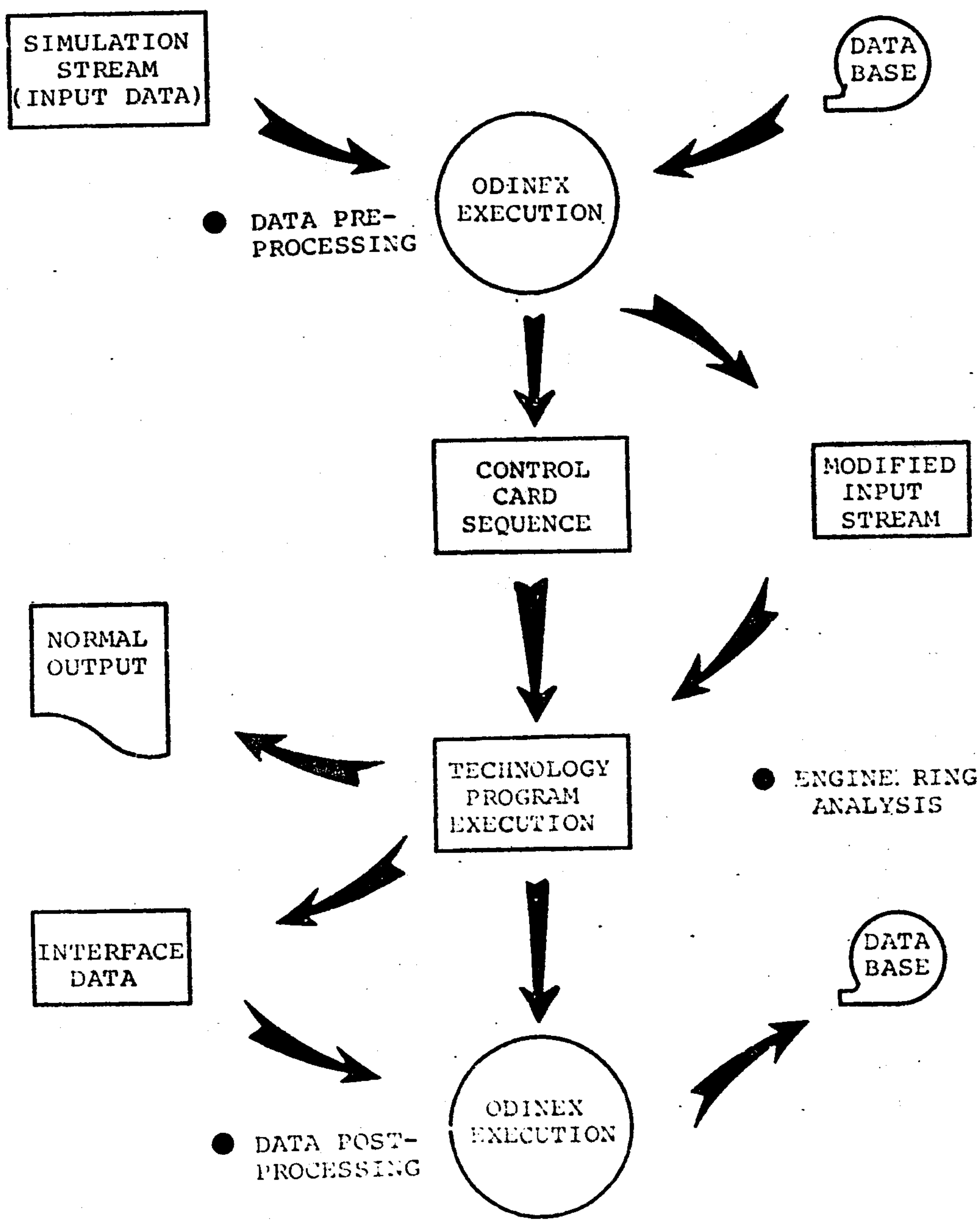


FIGURE B-1 SCENARIO FOR THE EXECUTION OF ONE TECHNOLOGY PROGRAM.

input stream. The control card sequence provides instructions to the computer for locating and executing the requested technology program. The technology program reads the modified input data, generates the normal output and may generate an interface data file of potential data base information. The next execution of ODINEX merges the interface data file with the data base, then repeats the entire cycle for the next technology program in the simulation stream. Instructions can be provided for looping backward or forward in the simulation stream.

Complete instructions for use of the ODINEX language elements in conjunction with the basic ODIN components are presented herein. In addition, an interface technique is described for allowing any program in the synthesis to update the data base. The technique does not influence the stand-alone operation of the program. More details on syntactical construction of the language elements may be obtained from reference 8.

Language Structure and Usage

The language for constructing a simulation stream consists of control directives and communication commands which are summarized in figure B-2. Each language element is delimited by a pair of apostrophes (') to distinguish ODINEX input from the input data and ~~execution control cards~~ in the simulation stream. For example, in the language element,

'NAME.......'

NAME is the name of the command or directive (i.e. CREATE or ADD). The remaining information within the delimited region is ancillary to the specific command. The executive program scans the simulation stream for delimited information and performs the function specified therein.

The distinction between the language types lies in the functions which each performs. Control directives are used for controlling the execution sequence of the library programs. Directives are included which execute control card sequences, define branch points in the simulation stream and provide branching instructions. Data base printing is also accomplished through control directives.

The control directives CREATE, UPDATE and EXECUTE specify the preprocessing of a user data file which follows the directive. In these cases a user end-of-file (*EOF starting in column 1) is required to flag the end of the data. All communication commands must be placed within one of the user data files described above.

* 'CREATE name'

File handling directive for initializing named data bases.

CCDATA is a special name used for the control card data base.

'DESIGN name'

Control directive defining a point in the execution sequence to which control may be transferred.

'END'

Control directive which flags the end of a simulation.

* 'EXECUTE name'

A directive for executing a prestored sequence of control cards by name.

'IF name.OP. name'

Condition for branching. Any number of conditions defined by the named data may be specified on cards after a LOOP TO directive. If more than one condition is specified, any one of the conditions satisfied will trigger the branch instruction. OP is a conditional operator (LT,LE,EQ,GE,GT).

'LOOP TO name'

Branching instruction referring to a design name. It can be conditional or unconditional.

'PRINT name'

Directive for printing data bases.

'RESTART'

Causes the use of an existing data base. The data base must have previously been defined and stored.

* 'UPDATE name'

File handling directive for updating named data bases.

* Data followed by user end-of-file (*EOF) is required.

FIGURE B-2 ODIN EXECUTIVE LANGUAGE.
(A) CONTROL DIRECTIVES.

'A '

Used to replace data base names and delimiters on an input card with data base information.

A may be a scaler or vector data base entry of real, integer, hollerith or logical type.

A may be an arithmetic combination of real or integer data base variables, array elements, arrays or constants.

'ADD A = B, ...'

Used to create a new data base entry or alter existing data base entries.

A is a new or existing entry, scalar or element of an array.

B is the update information which can be real, integer, hollerith or logical constants, variables or arithmetic combinations thereof.

'. comment'

Used for placing descriptive information in the data stream. ODINEX replaces the comment and associate delimiters with blanks in the applications program data deck.

'DEFINE A = n, description,'

Used to define new or existing entries and reserve space in the data base. No data is entered.

A is the new or existing data base entry.

n is the desired number of data base locations. It is ignored if the entry exists, the default is 1 if omitted.

description is the hollerith information associated with the variable A.

'INSERT F = n/m'

Used for inserting subsets of BCD files within the modified input stream at the location of the command.

F is the system file name.

n is the starting record.

m is the ending record (m = EOF identifies the end-of-file)

FIGURE B-2 ODIN EXECUTIVE LANGUAGE.
(B) COMMUNICATION COMMANDS.

All control directives must be placed outside the user file boundaries. The total simulation is terminated with the END directive. A typical simulation involving two programs is illustrated below:

```
'CREATE DBASE'
(DATA)
*EOF

'EXECUTE PGMA'
(DATA)
*EOF

'EXECUTE PGMB'
(DATA)
*EOF

'END'
```

In the above simulation stream the executive program will perform the indicated operations in three steps:

1. Create DBASE and enter the information indicated by (DATA).

2. Construct a control card sequence for the execution of PGMA, preprocess the information indicated by (DATA) and construct a modified input stream from it. Provide the linkage to the control card sequence.

3. Perform the operations of step 2 for PGMB.

The simulation stream may be augmented by conditional branching using the control directives:

```
'DESIGN POINT1'
 .
 .
 .
'LOOP TO POINT1'
'IF V1.LT.V2'
```

The design directive establishes a point in the simulation stream where control may be transferred. The LOOP TO directive directs the actual transfer and the IF directive provides conditions under which the transfer will be performed.

The communication commands provide the means by which data base storage and retrieval functions can be performed. They command the transfer of information from the analyst to the data base, from the data base to the technology programs and from the technology programs to the data base. In the transfer of information from the analyst to the data base, the ADD, DEFINE and INSERT commands are used. A comment command is also available for annotating data. In the transfer of information from the data base to the applications program, the replacement command is used. Data from the data base is transferred by name to the input stream of the technology programs. In the transfer of information from the technology programs to the data base, a special extension of the ADD command is employed. An interface file of ADD-like commands can be generated by any technology program for later processing by the ODIN executive program.

Construction and Maintenance of Data Bases

Data base construction and maintenance are accomplished by the use of the three control directives, CREATE, UPDATE and RESTART. The CREATE directive creates a new data base which is stored on a temporary disk file by the same name. By permanently storing the data on the disk file, the user may access the information at a later time. The use of the UPDATE directive in conjunction with previously created data bases, allows the user to modify the data contained therein. The RESTART directive specifies the use of an existing data base as is.

The basic constuction of data bases is similar but varies in total size and specific characteristics which can be specified by the user at creation time. Data bases consist of two distinct parts, a free storage array of packed information where the actual data is stored and a directory of names and pointers to the data in the free storage array. The directory and the data base have certain attributes which are assigned by the CREATE directive.

```
'CREATE name, attribute = value, attribute = value'
(DATA)
*EOF
```

Any or all of the attributes may be specified by the user as follows. If not specified, the default values will be used.

Attributes	Default	Description
LTOTAL	1024	Total number of computer words allocated to the specified data base.
NWORD	2	Number of computer words per data base entry.
DIRLEN	51	Total number of directory entries allocated to the data base.
LENDES	5	Number of words of descriptive information associated with each entry in the data base.

Each entry in the directory is the name of a data element or the name of an array of data elements. Up to DIRLEN entries may be made. Each entry may have a short description (LENDES-2) stored with it. Each data element occupies NWORD computer words. In establishing the size of the data base (LTOTAL), the user must consider the total number of elements of data desired in the data base, the directory length, the desired length of description and the number of words per data element. If NELMT is the total number of elements, then:

```
LTOTAL = DIRLEN*(LENDES + 3) + NELMT*NWORD
```

After a data base is initially established, the UPDATE directive may be used for adding or modifying the information contained therein. The form of the directive is:

```
'UPDATE name'
(DATA)
*EOF
```

Attribute parameters which are established once by the CREATE directive may not be specified by the UPDATE directive. If the use of the existing data base without modification is desired, the directive:

```
'RESTART'
```

is used. No attributes may be specified and no data may be added.

Design Data Base. - The basic communication command for entering information is the ADD command. It permits a variable name and value or values to be placed in the data base:

```
'ADD name = value, value,'
```

If the name is a new entry, any number of values may be added. The number of values associated with the name when it is first added is also the number of locations reserved in the data base for that information. Later modifications to the information can not create more data base space. The values may be real, integer, hollerith or logical. A single ADD command may be used for creating or updating many information sets.

```
'ADD V1 = 25., V2 = 30, V3 = ALPHA, V4 = .TRUE.,
      A = 10., 15., 20., 25., I = 4, 5, 6
     V4 = 5 * 0.,'
```

The data type is defined by the input data type. Upon retrieval, the data type is determined by the characteristics of the stored data. The format of the ADD statement is patterned after the FORTRAN NAMELIST feature and indeed has many of the same characteristics and usage rules. For example, all name/value sets are separated by commas (,); all elemental values of an array are separated by commas; the entire statement (command) is delimited. In the case of NAMELIST, the delimiter is a dollar ($) sign; in the case of ADD commands, the delimiter is ('). The rules for entering hollerith information and multiple constants differ from NAMELIST. For details, see reference 8.

The ADD command has additional capability not present in NAMELIST. The value associated with the ADD name may be a previously defined data base variable name:

```
ADD V1 = V2,
```

The effect of the above command is to transfer the information associated with V2 to the data base space assigned to V1. V1 may or may not exist prior to the ADD command. If V1 did not exist, space will be created in the data base as the information is transferred. If V1 did exist, then the information in V1 will be replaced by the information in V2. The transfer of information from one data base location to another is <u>generally limited to scalar quantities</u>. The ADD command may also be used for combining existing data base information with other data information or constants:

```
'ADD V1 = V2 * V3,'
```

The operation illustrated above indicates a multiplication of the two numbers on the right side of the equal (=) sign prior to transferring the resulting information to the space allocated to V1. Any algebraic operator may be employed as follows:

- + addition
- - subtraction
- * multiplication
- / division
- ** exponentiation

More than one operation may be performed on the right side of the equal (=) sign.

'ADD V1 = V2 + V3 * K,'

Up to ten operations may be performed within a single ADD command. The operations start from the equal sign and progress to the right. The first variable is combined with the second. The result of that operation is combined with the third. The result of that operation is combined with the fourth, etc. It should be noted that the hierarchy or order of the operations <u>does not conform to FORTRAN arithmetic</u>. For example, in the above illustration, V3 is added to V2, then the sum is multiplied by K.

If the data base were so defined, the directory may contain space for storing descriptive information. Depending on the value of CREATE parameters, LENDES space is reserved for each name entered equivalent to:

LENDES -2

The default value of 5 provides three computer words (30 characters) for descriptive information. The DEFINE command is used to store the descriptive information in the directory:

'DEFINE name, description,'

DEFINE name = value, description,'

Name is a new or existing data base entry. If the name exists, the description is added. If not, the name and description are added. If the name is a new entry, then the value may be used to reserve space in the data base for data elements to be added later. The absences of value on a new entry results in the reservation of space for a single data element.

Control Card Data Base. - The control card data base, CCDATA, is a special data base used for the storage of control card sequences representing computer instructions for retrieving and executing technology programs. The creation of CCDATA is accomplished as follows:

```
'CREATE CCDATA, NWORD = 8, LENDES = 4'
(CONTROL CARD DATA)
*EOF
```

The value of NWORD is usually 8 (80 characters) representing the width of a tab card. The width can be reduced if the stored control card representatives all fall within the width specification. The parameter LENDES is usually 4 (the minimum value) since descriptions are usually not necessary or can be specified on the control cards themselves. LTOTAL and DIRLEN may be selected on the same basis as described for the design data bases.

The method used to enter control card sequences into CCDATA deviates from that employed for entering data into the design data base. Each control card sequence is assigned a name which is specified on the first card. The control card sequence associated with the name continues until another name is encountered:

```
'name1 =
(CONTROL CARD SEQUENCE)
name 2 =
(CONTROL CARD SEQUENCE)
'
```

The CCDATA entries are delimited by apostrophes (') immediately before the first name (no embedded blanks) and on a separate card following the last control card entered. A control card sequence may contain any correctly formatted information but must include the following control card information:

1. The first card must specify a program retrieval if necessary or a system comment card.

2. The execution of one or more technology and/or utility programs.

3. A control card that specifies the re-execution of the execution program.

A sample CCDATA entry for the CDC 6600 at LRC is illustrated below:

```
PGMA =
FETCH,A0000,,BINARY,,OVL.
(OTHER PRE-EXECUTION CONTROL CARDS)
OVL,MODIN,OUT,NMLIST.
(POST-EXECUTION CONTROL CARDS)
CCLINK,ODINEX.
```

The card beginning with FETCH retrieves the program called OVL from permanent storage (data cell). The card beginning with OVL specifies the execution of OVL. Other file names on that card are:

MODIN - Modified Input Stream
OUT - Normal Output from OVL
NMLIST - Interface File

The card beginning with CCLINK specifies a system level linkage to the next ODINEX execution.

The storage of CCDATA for future use requires the storage of the system file of the same name. The ODIN system is designed to retrieve the CCDATA file at the beginning of each accumulation. CCDATA may be updated at that time. Updating the control card data base is similar in format to the original creation:

```
'UPDATE CCDATA'
(CONTROL CARD DATA)
*EOF
```

As with the design data base, data base parameters may not be re-specified on the UPDATE directive. Updates to the control card sequences are made on an one-for-one replacement cards starting with the first card specified and continuing for as many cards as are specified.

```
'name(n) =
(CONTROL CARD SEQUENCE)
'
```

The value, n, specifies the starting card, care must be taken to avoid exceeding the number of cards originally placed in the data base since space in existing sequences can not be created. However, new sequences may be entered.

The RESTART command has not significance for control card data bases since the system is designed to provide continuous availability of CCDATA.

Construction of Simulation Streams

There are three major considerations in the construction of design simulation streams:

1. Establish a design data base and control card data base for use in the simulation.

2. Construct the desired design sequence.

3. Provide data base interfaces within the input data sets for the technology programs.

Data bases are established at the beginning of the simulation on an one-time basis using the techniques discussed above. The design data base is defined first by use of a CREATE, UPDATE or RESTART directive. CCDATA may then be created or updated as required. The control card data base should contain all of the control card sequences necessary to the execution of the simulation. Many combinations can be conceived but the following sequence illustrates the construction of a new design data base and updating of the existing control card data base:

```
'CREATE DBASE'
(DATA)
*EOF
'UPDATE CCDATA'
(DATA)
*EOF
{SIMULATION DATA}
'END'
```

<u>Execution of a Sequence of Technology Programs</u>. - Now consider the problem of sequential execution of one or more technology programs using the ODIN system. Assume the following control directives:

```
'EXECUTE PGMA '
(DATA)
*EOF
```

'EXECUTE PGMB '

(DATA)

*EOF

'EXECUTE PGMC '

(DATA)

*EOF

The function of ODINEX is simply to retrieve the control cards sequences for PGMA, PGMB and PGMC from the control card data base and queue them for execution by the computer operating system. The operating system performs the various control card functions contained in the sequences. ODINEX also pre-processes the data associated with each program and constructs a modified input stream. The pre-processing function will be discussed later. The physical linkage between the control card sequences of the various applications programs and ODINEX is a CDC 6600 operating system utility program, CCLINK. CCLINK is executed at the end of each control card sequence in CCDATA to link the technology program sequences just executed to the ODINEX program sequences. The linkage is sustained by ODINEX until an END directive is encountered. Details of the operation of the example utility CCLINK are contained in reference 8. Similar utility programs are usually available for other computer systems.

Conditional Branching. - The ODINEX program can direct the transfer of control forward or backward in the simulation stream by the use of conditional branching instructions. This capability is achieved by the following control card directive language elements:

'DESIGN name'

'LOOP TO name'

'IF V1.LT.V2'

The DESIGN directive establishes an identifier name in the execution sequence where control may be returned (or skipped to). The LOOP TO directive points to the identifier name to which control is to be transferred. The IF directive is a conditional operator based upon design dependent logic. If absent, the LOOP TO directive is a mandatory branching command. V1 and V2 represent data base values or numeric constants which may be used for establishing the branching conditions of the applications programs.

The language elements permit a complicated system of analysis loops to be constructed for satisfying a variety of matching constraints.

The IF tests employed encompass the standard set of six tests in FORTRAN; although the form of the ODINEX control directive language test differs in form to that of FORTRAN. The six tests are:

'IF V1.LT.V2'	IF (V1<V2)
'IF V1.GT.V2'	IF (V1>V2)
'IF V1.LE.V2'	IF (V1≤V2)
'IF V1.GE.V2'	IF (V1≥V2)
'IF V1.EQ.V2'	IF (V1=V2)
'IF V1.NE.V2'	IF (V1≠V2)

As noted previously V1 and V2 are constants or variables constructed in the design data base through use of the ADD command or generated by any technology program in the synthesis and passed to the design data base (the method to be discussed later).

The ability to select alternative program execution paths based on design dependent logic as illustrated below:

```
..............
'EXECUTE PGMA'
'DESIGN POINTA'
'EXECUTE PGMB'
'LOOP TO POINTB'
'IF V1.EQ.V2'
'IF V3.LT.V4'
'EXECUTE PGMC'
'LOOP TO POINTA'
'DESIGN POINTB'
'EXECUTE PGMD'
..............
```

The above control directive sequence defines the execution of PGMA and PGMB with a conditional loop (in this case, skip) to POINTB. If neither of the IF conditions is satisfied, PGMC is executed followed by a mandatory loop back to POINTA.

Retrieval of Design Information

The retrieval of design information from the data base is accomplished by two basic communication commands:

1. Replacement Command.

2. INSERT Command.

The communication commands are strategically placed into the technology program input data sets. The augmented data sets form skeletonized input for the technology programs. Each data set is headed by the EXECUTE control directives corresponding to the appropriate technology programs. Collectively they form the design simulation stream described above.

Upon execution of the ODINEX program, the data sets are scanned for communication commands which must be delimited by apostrophe pairs (') for identification. Each encounter with delimited information causes ODINEX to perform the indicated replacement or insertion action. The result of all encounters within the skeletonized input for each technology program is a modified input stream which can be processed by the technology program.

<u>Replacement Command</u>. - Data base information may be entered into the technology program data sets by means of <u>delimited data base variable names</u> entered in the precise location where the requested data is to be placed. The delimiters define the field width for simple replacement (i.e. a single data element). The ODINEX program will replace the variable name and delimiters by its data base value and rewrite the card image on the modified input file. Input formats such as NAMELIST-like inputs and rigidly formatted input can be accommodated by the procedure. For example, in a true NAMELIST input, a data base variable would be entered as follows:

```
NAM1 = 'V1 ',
```

NAM1 is the name of the NAMELIST variable. V1 is the data base name. The delimiters specify the field width to be employed in replacing the data base name with the corresponding data base value. Similarly for a formatted input where data normally appears on a card within a specified range of columns, the data base replacement procedure is simply:

```
'V1    '
```

The delimiters are placed at the appropriate card columns defining the field for the data element. The data base name is placed within the defined field adjacent to the first delimiter.

Additional capability is available when namelist input is used by the technology program. Entire arrays may be transferred to the input stream using the following procedure:

NAME = 'VARRAY'

where VARRAY is a data base array name. If the data in VARRAY contains more data than can fit on one card record, additional 'cards' are created and placed in the modified input stream to accommodate the excess data.

Data base variables and constants may be combined much like the capability described for the ADD command above. For example, the operation:

'V1 * V2'

illustrates how the product of V1 and V2 may be used as the replacement value of data base variable. The product never resides in the data base but only in the modified input stream. V1 must be a data base variable name but V2 may be a data base variable or constant. More than one arithmetic operation may be performed within the delimiters:

'V1 + V2 * V3'

Up to ten operations may be performed within a single command. The hierarchy of operations is the same as that described for the ADD command.

In general, array elements may also be used in the replacement command:

'V1(5)'

or

'V1(5) * V2(6)'

One exception from this capability is in dealing with the first element of an array.

'V(1)'

The above command specifies the replacement of an entire data base array and is therefore not an acceptable command for replacing the first element of an array. The dilemma may be avoided by use of the following two cards:

'ADD NEW = V1(1)'

'NEW'

The ADD command defines a new data base variable which will contain the element V1(1). The replacement command will place the variable NEW (i.e. V1(1)) in the modified input stream. In general, all ADD command capability described under data base construction is applicable when placed in the skeletonized input of a technology program. The ADD command instructions are performed but the 'card' which contains the command and is not transferred to the modified input stream.

A special feature of the replacement command is the element-by-element combination of entire arrays or the combination of arrays with constants. As an illustration, consider the example:

'V1 * V2'

where the data base variables V1 and V2 are arrays. The above command specifies the element-by-element multiplication of the arrays. If one array has fewer elements than the other, the combining of elements ceases after the shorter array is exhausted and the res' of the longer array remains unchanged. The variable V2 may be a constant or data base name. Multiple operations may be performed using the hierarchical rules described above.

Insert Command. - Unlike the replacement command which extracts information from the design data base, the INSERT command transfers information sequentially from a formatted data card file. Starting and stopping positions within the file may be specified as follows:

'INSERT name1 = n/m, name2 = j/k'

The effect of the command is to search the named system files, name1 and name2, for integer card numbers n through m and j through k. If the card numbers are omitted, the entire named file is transferred to the modified input stream. The end-of-file may be specified for the named file by replacing m or k with the character string, EOF. The card containing the INSERT command is not transferred to the modified input stream.

Comment Command. - In addition to previously described commands, there exists a special "command" for identifying data. The format is:

'. comment'

No action is performed as a result of this command. It is useful only as an identifier for other data. For example, consider a technology program which uses formatted input (i.e. numbers with no identifiers or names associated with them). The comment command may be used to identify the data elements within the input data stream. The effect of processing this command through ODINEX is simply the replacement of the command with blanks. If the resulting card image is entirely blank, then the card is not transferred to the modified input stream.

Report Generation

A special feature of the ODINEX program is its ability to produce user generated report formats augmented with data base information. The report generator is applied in the following manner:

'EXECUTE REPORT'

(REPORT DATA)

*EOF

The report data is formatted by the user to provide any descriptive information desired. The report data may contain data base information through the use of the communication commands described above. One example of card image in the report might read:

WEIGHT OF THE SYSTEM IS 'WGT' POUNDS

In the above illustration, WGT is a data base variable name. The ODINEX executive program replaces the data base name and delimiter 'WGT' with the information stored in the data base. The report is printed through the normal computer output channels. The insertion of a report in the simulation stream does not effect the normal sequence of events for the simulation. The report may contain carriage control characters in column 1 of the report data cards.

1 - Eject a Page before Printing.

0 - Skip a Line before Printing.

Any number of reports may be generated during a simulation. The format of the individual reports is tailored to the needs of the particular study. Once the format is established, it can become a permanent part of the simulation stream. Any of the features of the ODINEX language, including scaling and adding data base information, are used in a completely free field report format.

Technology Program Interface Requirements

Interfacing technology programs require the transfer of information from the technology program to the ODIN data base where it becomes available to other technology programs. There are generally two methods of interfacing technology programs for use in the ODIN system:

1. Use of existing output files from the technology program.

2. Generation of special files of information within the technology program.

Many technology programs generate files of information suitable for use with the ODIN system, either directly or through an interface program. The interface program obtains its input from a file generated by a technology program. The output will be a special data file which can be processed by the ODINEX program. An interface program is often the most convenient means of extracting data because of the uncertainty associated with the modification of many technology programs.

Whether the technology program is modified or an interface program is written, the generation of the special output data file uses similar procedures which involve little programming knowledge to accomplish.

Creating a Special Output File. - The objective is to generate a file which can be processed by the ODINEX program and which contains the pertinent design information. The special file is made available to ODINEX by the use of an additional file parameter on the program card (CDC 6600 only).

```
PROGRAM PGMA (INPUT, OUTPUT, TAPE78)
```

The TAPE78 file is internally generated by PGMA and later transferred to the data base by the ODINEX executive program. The file number is optional and may be any available unit. The mechanism by which the file is passed from the technology program to the ODINEX executive is referred to as file substitution. As an illustration of file substitution technique, consider the execution of PGMA above. The execution control card for PGMA would be:

PGMA,MODIN,OUT,NMLIST.

The above card specifies the execution of PGMA. Additionally, it specifies that the INPUT file for PGMA will be MODIN, the OUTPUT file will be system file name OUT and the TAPE78 file will be the system file name NMLIST. These files are addressed internal to PGMA by the names on the program card, but addressed externally by the system file name. ODINEX recognizes NMLIST as a file containing potential data base information and processes the data contained therein.

Format of the Special Output File. - NMLIST is interrogated by ODINEX for name oriented information in the following format:

$name name = value, value, $

Note the similarity between this format and the ADD command format described above. The only difference is the delimiters employed. Also, it is identical with the FORTRAN NAMELIST feature. In generating the NMLIST file within the applications program, the analyst has the option of using NAMELIST (in FORTRAN programs only) or simply simulating NAMELIST as follows:

1. Separate name and values with equal (=) signs (N = V1).

2. Separate name/value sets with commas (,) (N1 = V1, N2 = V2).

3. Separate values (as in an array) with commas (,) (N3 = V1, V2).

4. Delimit multiple name value sets with a delimited ADD like "command" ($name.........$).

Generally the name selected is one which is similar to the program name such as:

$PGMAO

for PGMA output. The advantage of using this naming convention is apparent in the actual data base construction. The name PGMAO is stored in the data base directory identifying which program or "command" last updated the value or values stored.

Excluded Names and Special Options

Generally, any name except control directive names, communication command names and design identifiers may be used for naming data base information. However, there are a few additional exceptions. The excluded names are listed below:

BUILD	Option for dynamic construction of the data base.
DIVIDE = chr	Symbol used for divide (/) in the operator directory.
DOLLAR = chr	Symbol used for DOLLAR ($) in the operator directory.
ELTIME = value	Total elapsed simulation time used in timing option (must have 7 locations reserved in the data base).
EQUAL = chr	Symbol used for equal (=) in the operator directory.
EXPON = chr	Symbol used for exponentiation (**) in the operator directory.
MINUS = chr	Symbol used for subtraction (-) in the operator directory.
MLTPLY = chr	Symbol used for multiplication (*) in the operator directory.
NOTEQL = chr	Symbol used for a data base delimiter (≠) in the operator directory.
INDUMP	Option to print the modified input for every program.
OUTDUMP	Option to print the special data base output file from each program.
PAGDMP	Option to print the internal string processing information.
PLUS = chr	Symbol used for addition (+) in the operator directory.

The above names, when stored in the data base, represent special options in ODINEX. The data base variables are stored with the ADD command (i.e. 'ADD BUILD'). If present in the data base, ODINEX will reset control characters or perform special functions as described in the following paragraphs.

The BUILD option is associated with the dynamic construction of the design data base by the applications programs. Two values have significance.

BUILD = 0

BUILD = 1

The zero value specifies that previously undefined variables from the previous program will not be stored. A value of one specifies that all information from the previous program will be stored in the data base regardless of its previous data base status. The value of BUILD may be changed from program-to-program by use of the ADD command:

'ADD BUILD = n'

in the input data of the applications program. The change in the BUILD option becomes effective for the program where it occurs and remains effective until changed again. The BUILD option only affects data coming from technology programs. It has no effect on ADD commands or other user communication commands.

DBDUMP is an option which specifies that the entire data base be printed after each applications program execution. It should be noted that this option is mandatory when selected. Selective printing of the data base is accomplished by the PRINT directive.

ELTIME is a timing option. It is selected by a defining ELTIME and reserving 7 locations in the data base.

'DEFINE ELTIME = 7, description,'

When selected, a timing routine will be activated. This routine monitors the individual and cumulative computer processing parameters for the applications programs. The parameters are identified on the printed output.

INDUMP is an optional data base name which specifies the printing of the modified input stream for the applications programs. The modified input stream represents the input file in exactly the form the applications program will read it. The INDUMP option is useful in the early phases of linking programs for debugging purposes.

OUTDMP is an optional data base name which specifies the printing of the special data base output file, NMLIST, which contains all the information available for entry into the data base. The OUT-DMP can be used to determine what information is available from a given applications program or simply as a debugging aid in the early phases of the analysis.

PAGDMP is an option available to the programmer for detecting errors within the ODINEX program. It has little use to the analyst using the system. It is mentioned only because PAGDMP is an excluded data base name.